Erschienen in: Pattern Recognition : 44th DAGM German Conference, DAGM GCPR 2022, Konstanz, Germany, September 27–30, 2022, Proceedings / Andres, Björn et al. (Hrsg.). - Cham : Springer, 2022. - (Lecture Notes in Computer Science ; 13485). - S. 230-245. - ISSN 0302-9743. - eISSN 1611-3349. - ISBN 978-3-031-16787-4
https://dx.doi.org/10.1007/978-3-031-16788-1_15

Improving Unsupervised Label Propagation for Pose Tracking and Video Object Segmentation

Urs Waldmann[1,2(✉)], Jannik Bamberger[1], Ole Johannsen[1], Oliver Deussen[1,2], and Bastian Goldlücke[1,2]

[1] Department of Computer and Information Science, University of Konstanz, Konstanz, Germany
urs.waldmann@uni-konstanz.de

[2] Centre for the Advanced Study of Collective Behaviour, University of Konstanz, Konstanz, Germany

Abstract. Label propagation is a challenging task in computer vision with many applications. One approach is to learn representations of visual correspondence. In this paper, we study recent works on label propagation based on correspondence, carefully evaluate the effect of various aspects of their implementation, and improve upon various details. Our pipeline assembled from these best practices outperforms the previous state of the art in terms of $PCK_{0.1}$ on the JHMDB dataset by 6.5%. We also propose a novel joint framework for tracking and keypoint propagation, which in contrast to the core pipeline is applicable to tracking small objects and obtains results that substantially exceed the performance of the core pipeline. Finally, for VOS, we extend the core pipeline to a fully unsupervised one by initializing the first frame with the self-attention layer from DINO. Our pipeline for VOS runs online and can handle static objects. It outperforms unsupervised frameworks with these characteristics.

Keywords: Label propagation · Unsupervised · Pose tracking · VOS

1 Introduction

Learning representations of visual and temporal correspondence is a fundamental task in computer vision. It has many applications ranging from depth estimation [16] and optical flow [2,4,12] to segmentation and tracking [38,40,50]. Many

We acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2117 - 422037984. The authors also thank Hemal Naik, Nagy Máté, Fumihiro Kano and Iain D. Couzin for providing the real-world pigeon data.

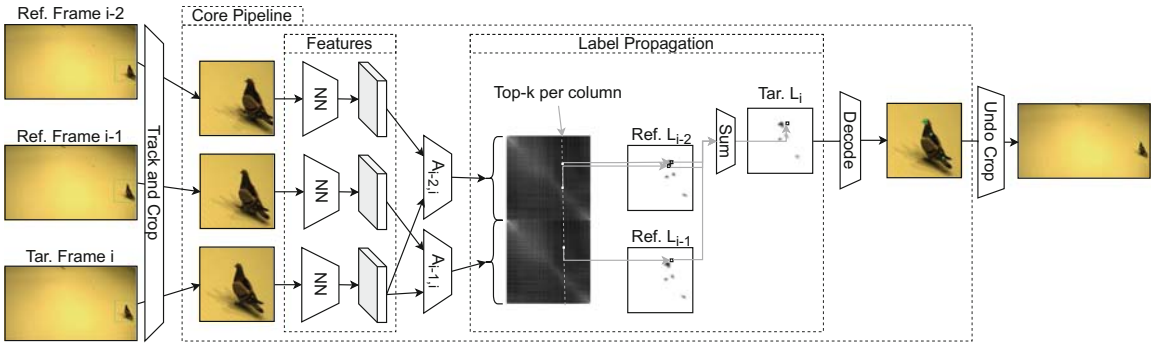

Fig. 1. *Overview of the joint tracking and label propagation pipeline, showing pigeon keypoint tracking as an example.* The core pipeline of our method uses deep neural network features to compute affinity matrices between a target frame and one or multiple reference frames. Labels for the target frame are computed by propagating the reference labels based on the combined affinity matrices. Finally, the predicted labels are decoded to coordinate values. The proposed extended pipeline shown outside the dashed box is necessary for tracking small objects. Here, an object tracker is used to locate the object of interest in each frame. This allows us to crop a region around the object and perform keypoint propagation at a much higher resolution, after which we undo the coordinate transform of the cropping for the decoded keypoints.

methods have been developed that can be split into two main categories [26], according to whether they learn correspondence on a pixel level [1,44,48,51,57] or a coarser level, i.e. region or object-based [12,27,30,43]. Recent approaches show that the two tasks can be modeled together with a single transformation operation [26,52].

The first networks learning representations for visual correspondence were trained in a supervised manner [4,11,12,39,42,46,49] and have some limitations. In particular, they do not generalize well to real-world scenes when trained on synthetic data, or rely on human annotations. That is why subsequently methods with less supervision have been preferred. Most of them are self-supervised or weakly supervised [13,22,23,26,29,52], and also reason at various levels of visual correspondence.

We focus on the notion of visual correspondence in tracking since it plays a key role in many applications, ranging from autonomous driving [6] to biology [15]. Categories of tracking differ in which kind of information is tracked throughout the frames. In Visual Object Tracking (VOT) [21], the typical aim is to track just the position or bounding box of an object throughout a video, given the location in the first frame. In contrast, Video Object Segmentation (VOS) [37] aims for a more fine-grained tracking, where we track segmentation masks on the pixel-level instead of bounding boxes. Somewhere in between, there is pose keypoint propagation, which tracks keypoints or a skeleton of an object. All of these tracking tasks can be subsumed within the framework of label propagation [48], on which we focus in this work.

Many recent state-of-the-art approaches for unsupervised label propagation employ similar methodology, in particular correspondence-based representation learning [3,13,22,23,26,52]. During inference, they also tend to do something sim-

ilar at the core, and compute a pixel-to-pixel affinity (similarity score) between adjacent frames, using this affinity function to propagate the label function. The computation of the affinity is predominantly based on top-k (cf. Sect. 3.2) and context frames (long-term vs. short-term, cf. Sect. 3.2). Further key parameters are feature, affinity and label normalization. Although using a similar methodology, the actual implementations of label propagation are quite different in the details. At the same time, these details are often not discussed at great length in the papers, as they can be found in the accompanying source code.

However, in experiments we find that these minor details in implementation can indeed have a very big impact on the performance, and that there are further improvements to be made which also contribute to a better accuracy. We therefore believe that a study of the effects of the details, in conjunction with a discussion on the best choice for particular implementations, can be a quite useful contribution.

Contributions. First, we present a pipeline for label propagation based on visual correspondence, which is built by carefully evaluating and selecting best practices, and improving upon previous work in important implementation details. This way, we outperform state of the art pipelines in terms of Percentage of Correct Keypoints ($PCK_{0.1}$) on the JHMDB [14] dataset by at least 6.5% in pose tracking. Second, we extend the pipeline to a joint tracking and keypoint propagation framework. This allows us to perform pose tracking for a novel data set for pigeons, which plays a role in our biological research on collective behaviour, and where the objects to be tracked are small compared to the frame size. With the proposed technique, we obtain excellent performance in pose tracking, exceeding the performance for humans (JHMDB) and substantially surpassing the core pipeline. Finally, for VOS, we present a pipeline which is fully unsupervised, initializing the first frame with the self-attention layer from DINO [3] that is trained without supervision. Here, if we disregard non-interactive post-processing, we outperform other unsupervised methods that do not rely on motion segmentation, which limits the method to moving objects. Both dataset and our code are publicly available at https://urs-waldmann.github.io/improving-unsupervised-label-propagation/.

2 Related Work

Unsupervised Label Propagation. Unsupervised label propagation has been studied extensively in recent years, for example to overcome the need for annotated training data. The key idea in Wang et al. [52] is to train with UDT-like [50] cycles, i.e. tracking backward and forward along a cycle in time. The inconsistency between start and end point serves as the loss function. The authors also demand cycle-consistency of a static track by forcing the tracker to relocate the next patch in each successive frame. To overcome momentary occlusions and ease learning in case of sudden changes in object pose, they also implement skip-cycles and learn from shorter cycles when the full cycle is too difficult. Lai and Xie [23] extend a framework [48] where tracking emerges automatically by learning the target colors for a gray-scale input frame. The authors add channel-wise dropout

and color jitter to the input. Li et al. [26] show that enforcing cycle-consistency is equivalent to regularizing the affinity to be orthogonal, i.e. they encourage every pixel to fall into the same location after one cycle of forward and backward tracking. Lai et al. [22] augment the architecture with a long- and short-term memory component to remember older frames: a two-step attention mechanism first coarsely searches for candidate windows and then computes the fine-grained matching. This ROI-based affinity computation additionally saves memory and computational overhead. Jabri et al. [13] use a Graph Neural Network [8,47] to learn representations of visual and temporal correspondence. The authors demand cycle-consistency for a walk on a space-time graph that is constructed from a video, where nodes are image patches and edges are only between neighbouring nodes. They include also edge dropout [41] and test-time adaptation. i.e. the model can be improved for correspondence by fine-tuning the representation at test time on new videos. In contrast to us, they need to manually initialize the mask of the first frame. Caron et al. [3] follow the experimental protocol in [13], but use different features. Our core pipeline is similar to UVC [26] but includes some improvements from STC [13] and CycleTime [52]. Further, we include new improvements to the keypoint label creation, add frame upscaling and object tracking [50] to improve the keypoint accuracy.

Video Object Segmentation. Yang et al. in [53] introduce an architecture where they exploit motion, using only optical flow as input, and train it without any manual supervision. They use L2 loss between input and reconstructed flow, pixel-wise entropy regularisation on inferred masks, multi-step flow I from multiple time steps as objects may be static for some frames, i.e. $\{I_{t\to t+n_1}, I_{t\to t+n_2}\}$, $n_1, n_2 \in \{-2,-1,1,2\}$, and a consistency loss. Yang et al. in [55] present an adversarial contextual model to detect moving objects in images. Their approach exploits classical region-based variational segmentation in addition to modern deep learning-based self-supervision. The authors encourage temporal consistency on the final masks and apply a CRF-based [20] post-processing on them.

In contrast to both works [53,55] we use an architecture based on correspondence that does not require object motion and can handle also objects which are static for longer time intervals. We are also able to propagate both masks and keypoints, while they can only propagate masks. In addition, we initialize the first frame with the self-attention layer from DINO [3] that is trained without supervision, so we do not depend on human annotation at all. Furthermore, in contrast to [55], our framework runs online, while they make use of offline post-processing.

3 Label Propagation

3.1 General Task

Label propagation is the task of propagating a number l of one or more label functions in a semantically plausible way trough a video. Given an initial set of label functions $L_1 \in [0,1]^{h\times w\times l}$ and a video $V = \{I_i | i \in 1,\ldots,N\}$ with frames

$I_i \in \mathcal{C}^{H \times W}$ in color space $\mathcal{C}$, the goal is to predict a new set of label functions $L_i \in [0,1]^{h \times w \times l}, i \geq 2$ for each subsequent frame in the video. Note that label functions are typically given at a lower resolution than the input images to match the feature extractor.

The common approach to this problem is based on training a feature extractor $\mathcal{F} : \mathcal{C}^{H \times W} \rightarrow \mathbb{R}^{h \times w \times d}$. It has to create location-aware features, i.e. there needs to be a spatial pixel-to-feature correspondence. The feature extractor is then used to compute pixel-wise frame-to-frame affinities which represent the semantic similarity, the larger, the stronger the semantic similarity. Let $F_i := \mathcal{F}(I_i)$ be the feature map for frame i and $\bar{F}_i \in \mathbb{R}^{hw \times d}$ be the linearized feature representation, where each row is a feature vector corresponding to one spatial location in the feature map. Then the affinity between frames I_i and I_j is given by $A_{i,j} = \bar{F}_i \, \bar{F}_j^{\,T} \in \mathbb{R}^{hw \times hw}$. Each entry in this affinity map corresponds to the semantic similarity of a pixel in frame I_i and another pixel in frame I_j.

To propagate a label function L_{i-1} by one frame – i.e. predicting L_i – the affinity matrix $A_{i-1,i}$ is computed. This allows us to compute the pixels of L_i from the correspondence $\hat{p}$ with highest affinity,

$$L_i = L_{i-1}\big(c(\hat{p})\big) \text{ with } \hat{p}(x,y) = \underset{q \in \{1,\dots hw\}}{\operatorname{argmax}} \left\{ A_{i-1,i}(\, q,\, c^{-1}(x,y)\,) \right\}. \tag{1}$$

Here, $c : \{1,\dots,hw\} \rightarrow \{1,\dots,h\} \times \{1,\dots,w\}$ is a coordinate translation function that maps the linearized coordinates in $\bar{F}$ to the corresponding spatial coordinates in F.

Choice of Feature Extractor. The feature extraction is an integral building block of label propagation. In this work we are not concerned with different strategies to build and train the feature extraction network, but look at the inference pipeline used for label propagation, so no new networks were trained. We use a ResNet18 [10] trained with the methodology of Li et al. [26] for pose tracking. Even though this method is no longer state of the art, we show that minor improvements to the inference pipeline can make the method competitive again. For our fully unsupervised segment propagation experiments, we rely on a vision transformer [18] trained with DINO [3]. In line with [3,13,26], we found that L_2 normalization of each feature vector is beneficial for the tracking performance (cf. 'No Feature Normalization' vs. 'Baseline' in Table 1).

3.2 Common Extensions

The general approach to label propagation is conceptually sound, but has some difficulties in the real world. Real-world data is noisy, the feature extractors are not able to compute exact semantic similarities and the compute resources are limited. Therefore, a myriad of extensions and modifications to the simple label propagation are made, which are different across methods. In our work, we analyze various implementations of current label propagation methods and notice that they tend to implement the same functionality in slightly different ways. We took this as an opportunity to look at the performance implications of such differences. This allows us to combine the best choice for each subtask to obtain higher-quality results than each method individually.

Affinity Top-K and Normalization. In Eq. (1) we have used the location of the maximum to select the propagated value. In theory, this is the optimal choice, but the noisy nature of the implementation can lead to outliers in the affinity matrix. To mitigate the impact of a single outlier, it is common [13,22,23,26,52] to select more than one similar location and aggregate the corresponding label function values. The aggregation of values differs from method to method. CycleTime [52] applies a softmax function to the affinity matrix columns and then aggregates the top k values with a weighted sum. The affinity scores act as the weights. In contrast, STC [13] selects the top k values first and then applies the softmax function afterwards. Further, UVC [26] and STC [13] have a temperature parameter in their softmax implementation that helps to sharpen or soften the output distribution. Many methods [3,26] also apply a network-specific transformation to the affinity values. Our implementation uses a top-k-based implementation with network-specific transformations for UVC and DINO. We perform the normalization after applying top-k, using softmax for UVC and a simple normalization for DINO. We found that these combinations work best for the respective networks. Results of ablation experiments concerning the number of reference locations k (top-k also compared against a pure argmax approach) and the normalization are shown in Table 1.

Local Affinity. Another measure to prevent implausible solutions is the use of local affinity [13,23,26]. An object in a video can only travel a limited distance between frames. The actual number of pixels depends on parameters like movement speed, camera motion and frame rate. Assumptions about the maximum displacement of an object between frames allow us to consider only a fixed-size neighborhood around each location for propagation. The benefits of this approach are two-fold. For one, the restriction to plausible movements in the image prevents some undesired solutions that would lead to discontinuities in the predicted label functions. This restriction leads to better results (cf. Table 1). Secondly, a consistent local neighborhood implementation brings potential for large performance improvements and memory reduction. Because only a small area around each pixel is required, the affinity matrix does not have to be computed entirely, and the memory requirement exhibits a linear instead of a quadratic growth in the number of pixels.

Context Frames. Videos commonly contain frames of varying quality, thus it proves beneficial to use more than one reference frame at once. In this work we compared two approaches to using such context frames: First, we analyzed the results of individual propagation from each context frame and subsequent aggregation of the individual results. Secondly, we analyzed the joint propagation from all context frames at once, cf. Fig. 1. The first approach is used by [23,26,52] and simply averages the label function results of each context frame. The joint propagation used by [3,13] treats the affinity matrices of each context frame as one big matrix with columns containing positions in all context frames. The propagation is then performed by selecting the top-k values from the columns of

this joint affinity matrix. This joint approach allows to skip bad frames altogether and instead select more values from higher-quality frames. We found that the second approach works substantially better (cf. 'Batched Label Propagation' vs. 'Baseline' in Table 1). The context frames we use are the first frame and a fixed number of directly preceding frames (cf. Figs. 1 and 2 and Tables 2 and 3 in our supplemental material). Using the first frame as context is advisable as the label functions for this frame are given. Hence, they are correct all the time which helps to avoid drift during longer tracking.

4 Applications and Case Studies

The generic way of formulating label propagation enables the propagation of various kinds of information through a video with a single framework. Examples include masks, keypoints and textures. Here, we take a look at mask and keypoint propagation. For each of these tasks, we have certain information in our problem domain $\mathbb{P}$, which we have to map to our label propagation framework. Thus, we have to define an encoding function $E : \mathbb{P} \to [0,1]^{h \times w \times l}$ and a decoder $D : [0,1]^{h \times w \times l} \to \mathbb{P}$. Definitions of our encoders and decoders with their implementation details are in Sect. 1 of our supplemental material.

4.1 Pose Tracking

We implement our encoding and decoding functions for the pose tracking core pipeline (cf. dashed box in Fig. 1) in such a way that the keypoints are computed with sub-pixel precision (cf. Sect. 1.1 in our supplementary).

We also contribute a new finding related to the sub-pixel precision requirement of keypoint encoding and decoding. Somewhat surprisingly, we found that upsampling of the input frames before processing is actually of substantial benefit. Although this does not add any additional information, the increase in spatial resolution of the feature map reduces the precision loss during encoding and decoding and leads to an overall gain in performance (cf. 'Image Scale 320' vs. 'Baseline' in Table 1). In consequence, our final proposed pipeline includes this step.

4.2 Joint Tracking and Keypoint Propagation

Inspired by our particular use case of pigeons (cf. Fig. 4), we first track the object using [50] and then propagate the keypoints on the cropped images (cf. Sect. 4.1). This is our novel joint tracking and keypoint propagation framework (cf. Fig. 1). Details are provided in Sect. 1 in the supplementary material. We use this novel joint tracking and keypoint propagation pipeline only for pose tracking, not VOS.

4.3 Unsupervised Zero-Shot VOS

Encoding a set of $j = 1, \ldots, n$ masks $M_i^j \in \{0,1\}^{H \times W}$ for unsupervised VOS is relatively simple [22,23,26,52]. Details on this and our unsupervised mask initialization can be found in Sect. 1.3 in our supplemental material.

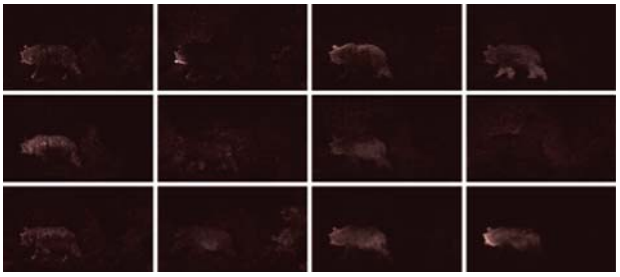

Fig. 2. Attention heads of the last attention layer of the CLS token. The input image is the first frame of the *bear* sequence of DAVIS2017 [37].

Fig. 3. Ground truth mask (left), input frame (middle) and predicted mask (right) for the *bear*, *dog-agility* and *drift-straight* sequences of DAVIS2017.

Unsupervised Mask Initialization. Caron et al. [3] found that their unsupervised training method DINO results in a vision transformer with semantically relevant attention (cf. Fig. 1 in [3]). We use this finding together with the learned attention layer to perform fully unsupervised object segmentation. Note that so far, the only human intervention in our tracking pipeline is the initialization in the first frame. To get rid of it, we generate a mask prediction for the first frame from the last attention layer of the CLS token. Some of the attention heads focus on background objects (e.g. Fig. 2 second row, second and fourth image). Further, we observe that the attention heads seldom focus on the entire object, and are more likely to select a single feature of the object of interest. Hence, we employ a heuristic to select attention heads that are important by computing the median value per attention head and discarding all masks with a median value below a fixed quantile. The reasoning behind this selection is that attention focused on the background is spread out more than attention focusing on a single feature or object. Therefore, the median value is below average.

Next, we follow the steps of Caron et al. [3] to select pixel locations per attention head.

One disadvantage of the attention is that it is rather coarse. Thus, we improve the mask quality by performing CRF-based refinement which fits the mask prediction to the input image. These mask refinement steps are similar to the post-processing steps performed by some other label propagation methods [29,55].

Note that this approach only works for a single object, and can fail for objects that occupy only a small area of the image. This is due to the nature of our attention selection mechanism, where we combine multiple attention heads. Another caveat is the heuristic-based nature of this approach. Ideally, we would find a way to perform the attention head selection automatically. Still, the method shows that the use of attention for mask initialization already gives promising results as shown in Fig. 3.

Real-world dataset	$PCK_{0.1}$	$PCK_{0.2}$
Core pipeline	7.5%	25.1%
Joint pipeline	81.0%	97.5%
Synthetic dataset	$PCK_{0.1}$	$PCK_{0.2}$
Core pipeline	0.4%	1.0%
Joint pipeline	65.7%	89.1%

Fig. 4. *Results for pigeon tracking, showing the last frame of the sequence with the tracked keypoints. Left image*: crop of the real-world sequence. *Right image*: crop of the synthetic dataset. *Table*: comparison of the results for the core pipeline (Sect. 4.1) and the proposed joint tracking and keypoint propagation pipeline introduced in Sect. 4.2.

5 Experiments

In this section we present ablation studies and discuss results of our applications and case studies. For more ablation studies, additional qualitative and quantitative results, we refer to Sects. 2–4 respectively in our supplemental material.

5.1 Ablation Studies

Table 1. *Ablation study of joint tracking and keypoint propagation on the JHMDB test set.* We measure performance as the percentage of correct keypoints (PCK) [56]. Keypoints are correct if the distance to the corresponding ground truth keypoint relative to the object size falls within a threshold α. Our implementation of PCK is equivalent to the one used by UVC [26], using 60% of the bounding box diagonal as the baseline radius ($\alpha = 1.0$). PCK is shown at thresholds $\alpha = 0.1$ and $\alpha = 0.2$, while Δ_α denotes the absolute difference to the baseline configuration which includes all improvements.

	$PCK_{0.1}$	$\Delta_{0.1}$	$PCK_{0.2}$	$\Delta_{0.2}$
Baseline	65.8	–	84.2	–
No Feature Normalization	64.0	–1.8	82.5	–1.7
Affinity Top-1 (ArgMax)	59.8	–6.0	80.8	–3.4
Affinity Top-5	65.0	–0.8	83.8	–0.4
No Affinity Normalization	65.3	–0.5	83.9	–0.3
Affinity Softmax-Normalization	65.2	–0.6	83.8	–0.4
Unrestricted Local Affinity	65.6	–0.2	83.9	–0.3
Batched Label Propagation	61.6	–4.2	80.5	–3.7
No Subpixel-Accurate Labels	65.5	–0.3	84.0	–0.2
Image Scale 320	63.2	–2.6	82.1	–2.1

Pose Tracking. For our pose tracking framework, we perform various ablation experiments to ensure that our pose tracking pipeline works as intended, and analyze the impact of the different components and improvements in the implementation. The baseline configuration of the pipeline is specified in Sect. 4. The

Table 2. *Ablation study for different types of mask initialization.* Performance is measured as IoU using the first mask of DAVIS2016 as ground truth. Delta shows absolute difference to the baseline mask initialization.

Ablation	mean	Δ	median	Δ
Baseline	50.8	–	54.7	–
No med. pre-filter	49.1	−1.7	51.7	−3.0
No mask refinement	41.1	−9.7	38.3	−16.4
No med. post-filter	49.9	−0.9	53.9	−0.8

majority of experiments are summarized in Table 1. Besides the listed results, we also experimented with various local affinity sizes. The optimal size for our configuration is 12. Further, we looked at the influence of the number of similar locations to aggregate for label propagation (cf. affinity top-k, Sect. 3.2). Here, we found that increasing or decreasing the baseline value of 20 by 5 corresponds to a performance loss of roughly 0.2% $\text{PCK}_{0.1}$.

Unsupervised Mask Initialization. The baseline configuration is described in Sect. 4. All results using unsupervised mask initialization use the ViT_b/8 model trained with DINO [3]. We evaluate the performance of the mask initialization on its own by comparing our mask prediction and the ground truth mask of frames from the DAVIS2016 [35] dataset. The quality of the mask prediction is computed as the intersection over union (IoU) of the binary masks. Table 2 shows how the performance changes in response to the removal of mask refinement or one of the median filtering steps. Removal of the mask refinement step altogether leads to the biggest drop in performance. Intuitively, this makes sense, as we combine multiple coarse attention masks, thus the refinement can capture a lot of the details in the image. Mask refinement also requires most of the initialization time, thus we chose the number of iterations as 10 to strike a balance between result quality and speed. Going from 1 to 10 iterations improves the results by 5.2% but increasing the iterations further, up to 50, only gives an additional 0.5% while requiring significantly more time. For both the quantile and mass percentage, we found that changes from the baseline in either direction reduce the performance. Increases as well as decreases by 0.05 reduce the mean and median performance by values ranging from 0.5% to 6.9%. This trend continues for larger changes as well. Reducing the spatial resolution or transformer network size have a major negative impact on the performance. Using ViT_b/16 or ViT_s/8 decrease the mean IoU by 5.0% and by 11.9% respectively.

5.2 Label Propagation

Pose Tracking. We show that our modifications to the label propagation pipeline can achieve substantial improvements in tracking accuracy. Quantitative results on JHMDB [14] are given in Table 3 where 15 human keypoints are

Table 3. *Results for pose tracking on the JHMDB test set. Ours* denotes just the core pipeline, while *Ours + tracking* includes joint tracking and keypoint propagation introduced in Sec. 4.2. Values are given in PCK at thresholds 0.1 and 0.2, respectively.

	ClrPtr [48]	SIFTflow [27]	CycleTime [52]	mgPFF [19]	CorrFlow [23]	UVC [26]	STC [13]	Ours	Ours + trk	Supervised [54]
$PCK_{0.1}$	45.2	49.0	57.7	58.4	58.5	58.6	59.3	63.9	**65.8**	68.7
$PCK_{0.2}$	69.6	68.6	78.5	78.1	78.8	79.8	**84.9**	82.8	84.2	92.1

tracked. In terms of $PCK_{0.1}$, we achieve an improvement of 7.2% over UVC [26] and 6.5% over STC [13], respectively, the latter being the current state-of-the-art at the time of writing. However, for the less precise $PCK_{0.2}$, we outperform UVC but are behind STC. Since we use the exact network weights as UVC, it is possible that switching to the same feature extraction network as STC might make a difference here. We leave testing this hypothesis to future work.

Pigeon Keypoint Tracking. We use a newly created dataset with video material showing a pigeon. Seven keypoints located at the head, shoulders and tail are annotated. Frames have a resolution of 1080×1920 pixels and the pigeon occupies a region of roughly 200×150 pixels on average. A real-world video of 326 frames and a synthetically generated sequence (for details cf. our supplementary) with 949 frames form the dataset. Figure 4 shows results on these sequences. Clearly, with our tracking modification (cf. Sect. 4.2) our method can achieve a similar magnitude of performance as JHMDB whereas the core pipeline is not capable to handle such a scenario.

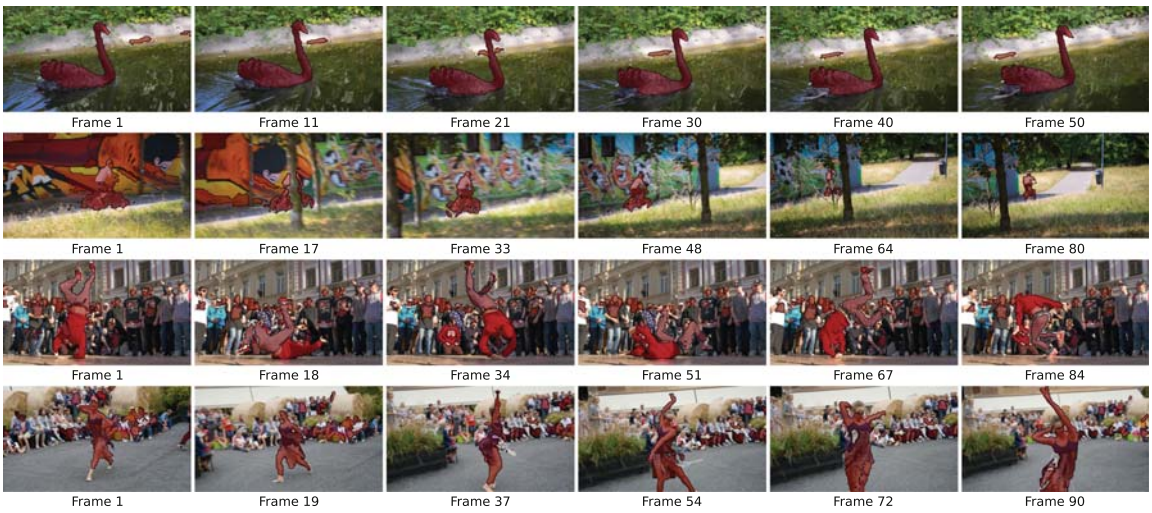

Fig. 5. *Qualitative results on DAVIS2016val with zero-shot inference.* The shown sequences are *blackswan*, *bmx-trees*, *breakdance* and *dance-twirl*, listed from top to bottom. The first frame shows the mask initialization result. The following frames were sampled over the entire video length maintaining even spacing in between.

Table 4. *Results for VOS on zero-shot DAVIS2016val. Supervision* refers to the the use of human annotations during training. Methods are considered unsupervised if no training is performed or no human annotations are used. Weak supervision refers to the use of human annotations without pixel-precision, e.g. bounding boxes or class labels. A method is considered *online*, if it processes videos as a stream of frames instead of using them all at once. The results have to be produced immediately, before the next frame is used. *Post-processing* refers to refinement steps after the actual segmentation, such as temporal smoothing or averaging of an ensemble of evaluation runs. Such techniques could be applied to other methods as well and they typically operate offline. *"Motion only"* marks methods that perform motion segmentation. These methods only work for moving objects, which limits their applicability. The results for methods other than our own are collected from the official DAVIS2016 results page [36] and the respective papers. For CIS [55], AMD [28] and 3DC-Seg [31], some measures were not reported, hence they are missing in the table. $*$: Results are available with a few frames delay.

Method	Online	Post-processing	Motion only	$\mathcal{J}\&\mathcal{F}_m$	$\mathcal{J}_m$	$\mathcal{J}_r$	$\mathcal{F}_m$	$\mathcal{F}_r$
Unsupervised								
MGR [34]	×	×	✓	44.0	48.9	44.7	39.1	28.6
KEY [24]	×	✓	×	46.3	49.8	59.1	42.7	37.5
MSG [32]	×	×	×	52.1	53.3	61.6	50.8	60.0
NLC [7]	×	✓	✓	53.7	55.1	55.8	52.3	51.9
FST [33]	×	✓	×	53.5	55.8	64.9	51.1	51.6
MuG [29]	×	✓	×	54.8	58.0	65.3	51.5	53.2
AMD [28]	✓	×	✓	–	57.8	–	–	–
CIS [55]	✓	×	✓	–	59.2	–	–	–
Ours	✓	×	×	59.8	61.6	71.3	58.0	63.3
MoGr [53]	✓	×	✓	64.7	68.3	79.5	61.1	72.1
CIS + pp [55]	×	✓	✓	–	71.5	–	–	–
Weakly supervised								
COSEG [45]	×	✓	×	51.1	52.8	50.0	49.3	52.7
MuG [29]	×	✓	×	58.7	61.2	74.5	56.1	62.1
Fully supervised								
FSEG [5]	✓	×	×	68.0	70.7	83.5	65.3	73.8
ARP [17]	×	×	×	73.4	76.2	91.1	70.6	83.5
MATNet [58]	✓	✓	×	81.6	82.4	94.5	80.7	90.2
3DC-Seg [31]	✓$*$	×	×	84.5	84.3	–	84.7	–

Table 5. *Z-VOS results on SegTrackV2.* The Z-VOS inference treats the dataset as a single-object dataset, i.e. all individual object masks are merged into one object. †: These methods use either offline post-processing or rely on some component with supervised training.

	MoGr [53]	CIS [55]	FST [33]	AMD [28]	SAGE [9]	Ours	MoGr†	CIS+pp†	NLC†[7]
$\mathcal{J}_m$	37.8	45.6	47.8	57.0	57.6	58.4	58.6	62.0	67.2

Unsupervised Zero-Shot VOS. Single-object Z-VOS results can be found in Tables 4 and 5 for DAVIS2016 and SegTrackV2 [25] respectively. On DAVIS2016, only motion segmentation methods (CIS [55] and MoGr [53]) achieve better performance. Motion segmentation has the disadvantage that static, non-moving objects cannot be segmented, whereas our method can also detect such objects. Incidentally, DAVIS2016 and SegTrackV2 contains only objects in motion, thus posing no problem for motion segmentation methods. Further, CIS uses extensive post-processing including an ensemble of models, temporal smoothing, and refinement with a CRF. Yang et al. [53] argument that a fair comparison should not use such post-processing, since it reduces real-world use of the methods. If we consider the results of CIS without the post-processing ($\mathcal{J}_m = 59.2$), our method surpasses this result by 2.4%. Some other methods also apply post-processing, as noted in Table 4. Additionally, some of the methods listed in the *unsupervised* section actually use some supervised components. NLC relies on supervised boundary detection and MoGr uses supervised optical flow to achieve the listed results. Compared to MoGr with unsupervised flow, we surpass their result of $\mathcal{J}_m = 53.2$ [53] by 8.4%. Qualitative single-object Z-VOS results on DAVIS2016val are in Fig. 5.

Qualitative results of fully unsupervised mask initialization instead are shown in Fig. 3 and the first column in Fig. 5. We can observe that our method has difficulties selecting a good mask for objects that appear at a very small scale, but for larger objects the results look visually good.

Inference Speed. A rigorous evaluation of the inference speed is impossible without similar implementations and data captured on identical hardware. This is out of scope for our work, but we give a rough expected performance in the supplemental material.

6 Conclusions

In this work, we perfom a careful study of the state-of-the-art in correspondence-based label propagation methods and present a pipeline composed of the best practices and with improvements in various implementation details. For pose tracking, the proposed pipeline outperforms previous state of the art in terms of $\text{PCK}_{0.1}$ on the JHMDB dataset by at least 6.5%, more precisely, 59.3% in [13] vs. 65.8% for our method. In addition, we propose a joint tracking and keypoint propagation framework which allows to accurately propagate keypoints in our new dataset for tracking of pigeons, which is not possible with the standard pipeline. Finally, for video object segmentation, we extend our pipeline to a truly unsupervised one by initializing the first frame with the self-attention layer from DINO that is trained with no supervision. Our method is online and can compete with state of the art pipelines. Note that while [53] achieves a higher score among the unsupervised pipelines, their framework (and others) is based on optical flow and does not work for static objects. Furthermore, while [55] can achieve a higher score with their post-processing framework, it is computationally intensive and could be applied to any of the methods.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: Fully-convolutional siamese networks for object tracking. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9914, pp. 850–865. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48881-3_56
2. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3024, pp. 25–36. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24673-2_3
3. Caron, M., et al.: Emerging properties in self-supervised vision transformers. In: ICCV (2021)
4. Dosovitskiy, A., et al.: Flownet: learning optical flow with convolutional networks. In: ICCV (2015)
5. Dutt Jain, S., Xiong, B., Grauman, K.: FusionSeg: learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. In: CVPR, pp. 3664–3673 (2017)
6. Ess, A., Schindler, K., Leibe, B., Gool, L.V.: Object detection and tracking for autonomous navigation in dynamic environments. Int. J. Rob. Res. **29**(14), 1707–1725 (2010)
7. Faktor, A., Irani, M.: Video segmentation by non-local consensus voting. In: BMVC (2014)
8. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS 2017, pp. 1025–1035 (2017)
9. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS, pp. 1025–1035 (2017)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
11. Held, D., Thrun, S., Savarese, S.: Learning to track at 100 FPS with deep regression networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 749–765. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_45
12. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: evolution of optical flow estimation with deep networks. In: CVPR (2017)
13. Jabri, A., Owens, A., Efros, A.: Space-time correspondence as a contrastive random walk. In: NeurIPS, pp. 19545–19560 (2020)
14. Jhuang, H., Gall, J., Zuffi, S., Schmid, C., Black, M.J.: Towards understanding action recognition. In: ICCV (2013)
15. Kays, R., Crofoot, M.C., Jetz, W., Wikelski, M.: Terrestrial animal tracking as an eye on life and planet. Science **348**(6240), aaa2478 (2015)
16. Kendall, A., et al.: End-to-end learning of geometry and context for deep stereo regression. In: ICCV (2017)
17. Koh, Y.J., Kim, C.S.: Primary object segmentation in videos based on region augmentation and reduction. In: CVPR, pp. 7417–7425 (2017)
18. Kolesnikov, A., Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. In: ICLR (2021)
19. Kong, S., Fowlkes, C.: Multigrid predictive filter flow for unsupervised learning on videos. arXiv preprint arXiv:1904.01693 (2019)

20. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with gaussian edge potentials. In: NeurIPS (2011)
21. Kristan, M., et al.: A novel performance evaluation methodology for single-target trackers. IEEE TPAMI **38**(11), 2137–2155 (2016)
22. Lai, Z., Lu, E., Xie, W.: Mast: a memory-augmented self-supervised tracker. In: CVPR (2020)
23. Lai, Z., Xie, W.: Self-supervised learning for video correspondence flow. In: BMVC (2019)
24. Lee, Y.J., Kim, J., Grauman, K.: Key-segments for video object segmentation. In: ICCV, pp. 1995–2002 (2011)
25. Li, F., Kim, T., Humayun, A., Tsai, D., Rehg, J.M.: Video segmentation by tracking many figure-ground segments. In: ICCV, pp. 2192–2199 (2013)
26. Li, X., Liu, S., De Mello, S., Wang, X., Kautz, J., Yang, M.H.: Joint-task self-supervised learning for temporal correspondence. In: NeurIPS (2019)
27. Liu, C., Yuen, J., Torralba, A.: Sift flow: dense correspondence across scenes and its applications. IEEE TPAMI **33**(5), 978–994 (2011)
28. Liu, R., Wu, Z., Yu, S., Lin, S.: The emergence of objectness: learning zero-shot segmentation from videos. In: NeurIPS, vol. 34 (2021)
29. Lu, X., Wang, W., Shen, J., Tai, Y.W., Crandall, D.J., Hoi, S.C.H.: Learning video object segmentation from unlabeled videos. In: CVPR (2020)
30. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: DARPA Image Understanding Workshop, pp. 121–130 (1981)
31. Mahadevan, S., Athar, A., Ošep, A., Hennen, S., Leal-Taixé, L., Leibe, B.: Making a case for 3D convolutions for object segmentation in videos. In: BMVC (2020)
32. Ochs, P., Brox, T.: Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In: ICCV, pp. 1583–1590 (2011)
33. Papazoglou, A., Ferrari, V.: Fast object segmentation in unconstrained video. In: ICCV, pp. 1777–1784 (2013)
34. Pathak, D., Girshick, R., Dollár, P., Darrell, T., Hariharan, B.: Learning features by watching objects move. In: CVPR, pp. 2701–2710 (2017)
35. Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., Sorkine-Hornung, A.: A benchmark dataset and evaluation methodology for video object segmentation. In: CVPR (2016)
36. Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., Sorkine-Hornung, A.: Official davis 2016 results list (2016). https://davischallenge.org/davis2016/soa_compare.html
37. Pont-Tuset, J., Perazzi, F., Caelles, S., Sorkine-Hornung, A., Arbeláez, P., Gool, L.V.: Davis challenge on video object segmentation 2017 (2017). https://davischallenge.org/challenge2017/index.html
38. Rafi, U., Doering, A., Leibe, B., Gall, J.: Self-supervised keypoint correspondences for multi-person pose estimation and tracking in videos. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12365, pp. 36–52. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58565-5_3
39. Ranjan, A., Black, M.J.: Optical flow estimation using a spatial pyramid network. In: CVPR (2017)
40. Smith, S., Brady, J.: Asset-2: real-time motion segmentation and shape tracking. IEEE TPAMI **17**(8), 814–820 (1995)

41. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(56), 1929–1958 (2014)
42. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: CNNs for optical flow using pyramid, warping, and cost volume. In: CVPR (2018)
43. Tao, R., Gavves, E., Smeulders, A.W.: Siamese instance search for tracking. In: CVPR (2016)
44. Thomee, B., et al.: Yfcc100m: the new data in multimedia research. Commun. ACM **59**(2), 64–73 (2016)
45. Tsai, Y.-H., Zhong, G., Yang, M.-H.: Semantic co-segmentation in videos. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 760–775. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_46
46. Valmadre, J., Bertinetto, L., Henriques, J., Vedaldi, A., Torr, P.H.S.: End-to-end representation learning for correlation filter based tracking. In: CVPR (2017)
47. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
48. Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., Murphy, K.: Tracking emerges by colorizing videos. In: ECCV (2018)
49. Wang, N., Yeung, D.Y.: Learning a deep compact image representation for visual tracking. In: NeurIPS, vol. 26 (2013)
50. Wang, N., Zhou, W., Song, Y., Ma, C., Liu, W., Li, H.: Unsupervised deep representation learning for real-time tracking. IJCV **129**, 400–418 (2021)
51. Wang, Q., Zhang, L., Bertinetto, L., Hu, W., Torr, P.H.: Fast online object tracking and segmentation: a unifying approach. In: CVPR (2019)
52. Wang, X., Jabri, A., Efros, A.A.: Learning correspondence from the cycle-consistency of time. In: CVPR (2019)
53. Yang, C., Lamdouar, H., Lu, E., Zisserman, A., Xie, W.: Self-supervised video object segmentation by motion grouping. In: ICCV, pp. 7177–7188 (2021)
54. Yang, L., Wang, Y., Xiong, X., Yang, J., Katsaggelos, A.K.: Efficient video object segmentation via network modulation. In: CVPR (2018)
55. Yang, Y., Loquercio, A., Scaramuzza, D., Soatto, S.: Unsupervised moving object detection via contextual information separation. In: CVPR, pp. 879–888 (2019)
56. Yang, Y., Ramanan, D.: Articulated human detection with flexible mixtures of parts. IEEE TPAMI **35**, 2878–2890 (2012)
57. Zhou, Q., Liang, X., Gong, K., Lin, L.: Adaptive temporal encoding network for video instance-level human parsing. In: ACM MM, p. 1527–1535 (2018)
58. Zhou, T., Wang, S., Zhou, Y., Yao, Y., Li, J., Shao, L.: Motion-attentive transition for zero-shot video object segmentation. In: AAAI, pp. 13066–13073 (2020)