arXiv:1903.07593v2 [cs.CV] 2 Apr 2019

Learning Correspondence from the Cycle-consistency of Time

Xiaolong Wang*
Carnegie Mellon University
xiaolonw@cs.cmu.edu

Allan Jabri*
UC Berkeley
ajabri@eecs.berkeley.edu

Alexei A. Efros
UC Berkeley
efros@eecs.berkeley.edu

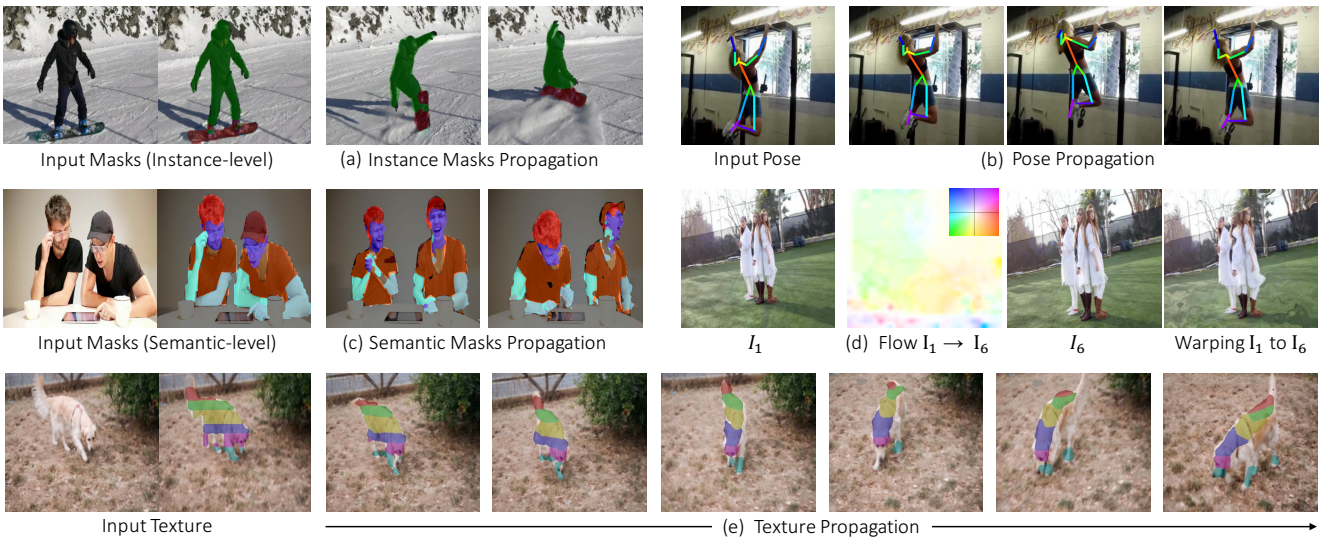


Figure 1: We propose to learn a representation for visual correspondence from raw video. Without any fine-tuning, the acquired representation generalizes to various tasks involving visual correspondence, allowing for propagation of: (a) Multiple Instance Masks; (b) Pose; (c) Semantic Masks; (d) Long-Range Optical Flow; (e) Texture.

Abstract

We introduce a self-supervised method for learning visual correspondence from unlabeled video. The main idea is to use cycle-consistency in time as free supervisory signal for learning visual representations from scratch. At training time, our model learns a feature map representation to be useful for performing cycle-consistent tracking. At test time, we use the acquired representation to find nearest neighbors across space and time. We demonstrate the generalizability of the representation – without finetuning – across a range of visual correspondence tasks, including video object segmentation, keypoint tracking, and optical flow. Our approach outperforms previous self-supervised methods and performs competitively with strongly supervised methods.[1]

1. Motivation

It is an oft-told story that when a young graduate student asked Takeo Kanade what are the three most important problems in computer vision, Kanade replied: "Correspondence, correspondence, correspondence!" Indeed, most fundamental vision problems, from optical flow and tracking to action recognition and 3D reconstruction, require some notion of visual correspondence. Correspondence is the glue that links disparate visual percepts into persistent entities and underlies visual reasoning in space and time.

Learning representations for visual correspondence, from pixel-wise to object-level, has been widely explored, primarily with supervised learning approaches requiring large amounts of labelled data. For learning low-level correspondence, such as optical flow, synthetic computer graphics data is often used as supervision [10, 22, 50, 62], limiting generalization to real scenes. On the other hand, approaches for learning higher-level semantic correspondence rely on human annotations [71, 19, 65], which becomes prohibitively expensive at large scale. In this work, our aim is to learn representations that support reasoning at various levels of visual correspondence (Figure 1) from scratch and without human supervision.

A fertile source of free supervision is video. Because the world does not change abruptly, there is inherent visual correspondence between observations adjacent in time. The problem is how to find these correspondences and turn

*Equal contribution.
[1]Project page: http://ajabri.github.io/timecycle

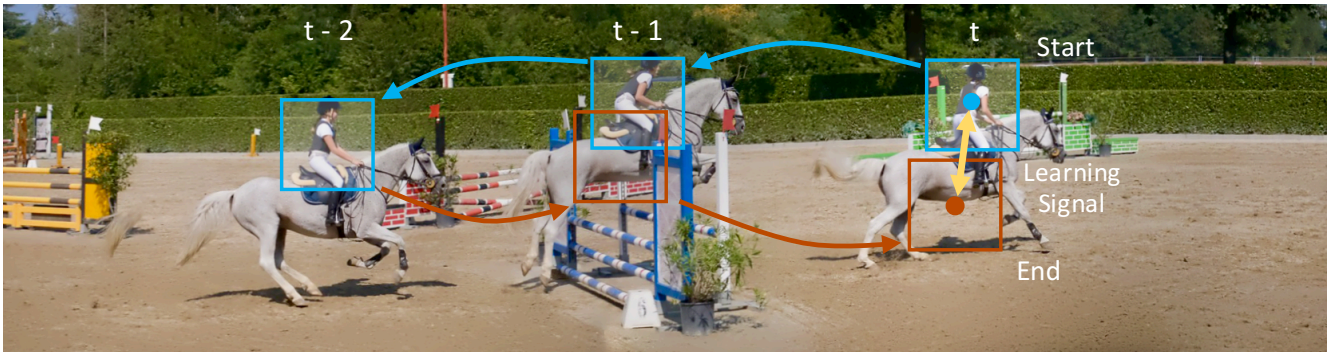


Figure 2: **A Cycle in Time.** Given a video, tracking along the sequence formed by a cycle in time can be self-supervised: the target is simply the beginning of the cycle. The yellow arrow between the start and end represents the differentiable learning signal.

them into a learning signal. In a largely static world observed by a stationary camera, such as a webcam trained on the Eiffel Tower, correspondence is straightforward because nothing moves and capturing visual invariance (to weather, lighting) amounts to supervised metric learning. In the dynamic world, however, change in appearance is confounded by movement in space. Finding correspondence becomes more difficult because capturing visual invariance now requires learning to track, but tracking relies on a model of visual invariance. This paper proposes to learn to do both simultaneously, in a self-supervised manner.

The key idea is that we can obtain unlimited supervision for correspondence by tracking backward and then forward (i.e. along a cycle in time) and using the *inconsistency* between the start and end points as the loss function (Figure 2). We perform tracking by template-matching in a learned deep feature space. To minimize the loss – i.e. to be *cycle-consistent* – the model must learn a feature representation that supports identifying correspondences across frames. As these features improve, the ability to track improves, inching the model toward cycle-consistency. Learning to chain correspondences in such a feature space should thus yield a visual similarity metric tolerant of local transformations in time, which can then be used at test-time as a stand-alone distance metric for correspondence.

While conceptually simple, implementing objectives based on cycle-consistency can be challenging. Without additional constraints, learning can take shortcuts, making correspondences cycle-consistent but wrong [88]. In our case, a track that never moves is inherently cycle-consistent. We avoid this by forcing the tracker to re-localize the next patch in each successive frame. Furthermore, cycle-consistency may not be achievable due to sudden changes in object pose or occlusions; *skip*-cycles can allow for cycle-consistency by skipping frames, as in Figure 3 (right). Finally, correspondence may be poor early in training, and shorter cycles may ease learning, as in Figure 3 (left). Thus, we simultaneously learn from many kinds of cycles to induce a natural curriculum and provide better training data.

The proposed formulation can be used with any differentiable tracking operation, providing a general framework for learning representations for visual correspondence from raw video. Because the method does not rely on human annotation, it can learn from the near infinite video data available online. We demonstrate the usefulness of the learned features for tasks at various levels of visual correspondence, ranging from pose, keypoint, and segmentation propagation (of objects and parts) to optical flow.

2. Related Work

Temporal Continuity in Visual Learning. Temporal structure serves as a useful signal for learning because the visual world is continuous and smoothly-varying. Spatio-temporal stability is thought to play a crucial role in the development of invariant representations in biological vision [83, 36, 77, 78]. For example, Wood [77] showed that for newborn chicks raised in a visual world that was not temporally smooth, object recognition abilities were severely impaired. Computational approaches for unsupervised learning have sought to leverage this continuity, such as continuous transformation learning [13, 70], "slow" feature learning [76, 91, 25] and information maximization between neighbouring patches in time [66]. Our work can be seen as slow feature learning with fixation, learned end-to-end without supervision.

Self-supervised Representation Learning from Video. Learning representations from video using time as supervision has been extensively studied, both as future prediction task [15, 60, 44, 42] as well as motion estimation [2, 25, 63, 38, 40]. Our approach is most related to the methods of Wang et al. [73, 74] and Pathak et al. [47], which use off-the-shelf tools for tracking and optical flow respectively, to provide supervisory signal for training. However, representations learned in this way are inherently limited by the power of these off-the-shelf tools as well as their failure modes. We address this issue by learning the representation and the tracker jointly, and find the two learning problems

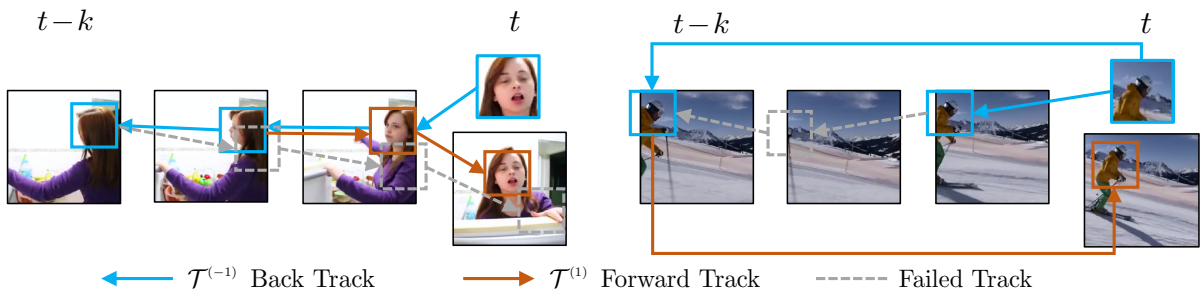


Figure 3: **Multiple Cycles and Skip Cycles.** Cycle-consistency may not be achievable due to sudden changes in object pose or occlusions. Our solution is to optimize multiple cycles of different lengths simultaneously. This allows learning from shorter cycles when the full cycle is too difficult (left). This also allows cycles that skip frames, which can deal with momentary occlusions (right).

to be complementary. Our work is also inspired by the innovative approach of Vondrick et al [69] where video colorization is used as a pretext self-supervised task for learning to track. While the idea is very intriguing, in Section 4 we find that colorization is a weaker source of supervision for correspondence than cycle-consistency, potentially due to the abundance of constant-color regions in natural scenes.

Tracking. Classic approaches to tracking treat it as a matching problem, where the goal is to find a given object/patch in the next frame (see [11] for overview), and the key challenge is to track reliably over extended time periods [57, 79, 1, 27]. Starting with the seminal work of Ramanan et al. [49], researchers largely turned to "tracking as repeated recognition", where trained object detectors are applied to each frame independently [3, 28, 80, 71, 19, 35, 65]. Our work harks back to the classic tracking-by-matching methods in treating it as a correspondence problem, but uses learning to obtain a robust representation that is able to model wide range of appearance changes.

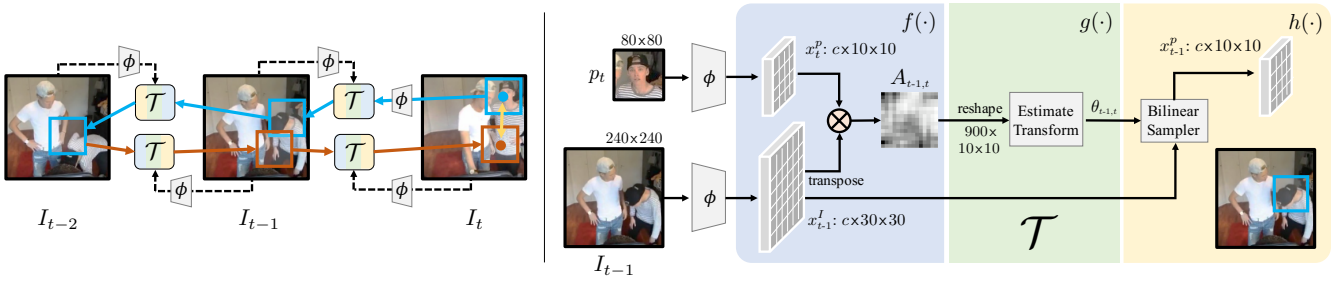
Optical Flow. Correspondence at the pixel level – mapping where each pixel goes in the next frame – is the optical flow estimation problem. Since the energy minimization framework of Horn and Schunck [20] and coarse-to-fine image warping by Lucas and Kanade [41], much progress has been made in optical flow estimation [46, 6, 61, 10, 22, 50, 62]. However, these methods still struggle to scale to long-range correspondence in dynamic scenes with partial observability. These issues have driven researchers to study methods for estimating long-range optical flow [56, 5, 55, 51, 52, 34]. For example, Brox and Malik [5] introduced a descriptor that matches region hierarchies and provides dense and subpixel-level estimation of flow. Our work can be viewed as enabling mid-level optical flow estimation.

Mid-level Correspondence. Given our focus on finding correspondence at the patch level, our method is also related to the classic SIFT Flow [39] algorithm and other methods for finding mid-level correspondences between regions across different scenes [30, 16, 87]. More recently, researchers have studied modeling correspondence in deep feature space [64, 33, 31, 17, 53, 54]. In particular, our work draws from Rocco et al. [53, 54], who propose a differentiable soft inlier score for evaluating quality of alignment between spatial features and provides a loss for learning semantic correspondences. Most of these methods rely on learning from simulated or large-scale labeled datasets such as ImageNet, or smaller custom human-annotated data with narrow scope. We address the challenge of learning representations of correspondence without human annotations.

Forward-Backward and Cycle Consistency. Our work is influenced by the classic idea of forward-backward consistency in tracking [57, 79, 1, 27], which has long been used as an evaluation metric for tracking [27] as well as a measure of uncertainty [1]. Recent work on optical flow estimation [43, 24, 68, 75, 45] also utilizes forward-backward consistency as an optimization goal. For example, Meister et al. [45] combines one-step forward and backward consistency check with pixel reconstruction loss for learning optical flows. Compared to pixel reconstruction, modeling correspondence in feature space allows us to follow and learn from longer cycles. Forward-backward consistency is a specific case of cycle-consistency, which has been widely applied as a learning objective for 3D shape matching [21], image alignment [87, 89, 88], depth estimation [86, 14, 84], and image-to-image translation [90, 4]. For example Zhou et al. [88] used 3D CAD models to render two synthetic views for pairs of training images and construct a correspondence flow 4-cycle. To the best of our knowledge, our work is the first to employ cycle-consistency across multiple steps in time.

3. Approach

An overview of the training procedure is presented in Figure 4a. The goal is to learn a feature space ϕ by tracking a patch p_t extracted from image I_t backwards and then forwards in time, while minimizing the cycle-consistency loss l_θ (yellow arrow). Learning ϕ relies on a simple tracking



(a) Training ϕ by End-to-end Cycle-consistent Tracking

(b) Differentiable Tracking Operation $\mathcal{T}$

Figure 4: **Method Overview**. (a) During training, the model learns a feature space encoded by ϕ to perform tracking using tracker $\mathcal{T}$. By tracking backward and then forward, we can use cycle-consistency to supervise learning of ϕ. Note that only the initial patch p_t is explicitly encoded by ϕ; other patch features along the cycle are obtained by localizing image features. (b) We show one step of tracking back in time from t to $t-1$. Given input image features x_{t-1}^I and query patch features x_t^p, $\mathcal{T}$ localizes the patch x_{t-1}^p in x_{t-1}^I. This operation is performed iteratively to track along the cycle in (a).

operation $\mathcal{T}$, which takes as inputs the features of a current patch and a target image, and returns the image feature region with maximum similarity. Our implementation of $\mathcal{T}$ is shown in Figure 4b: without information of where the patch came from, $\mathcal{T}$ must match features encoded by ϕ to localize the next patch. As shown in Figure 4a, $\mathcal{T}$ can be iteratively applied backwards and then forwards through time to track along an arbitrarily long cycle. The cycle-consistency loss l_θ is the euclidean distance between the spatial coordinates of initial patch p_t and the patch found at the end of the cycle in I_t. In order to minimize l_θ, the model must learn a feature space ϕ that allows for robustly measuring visual similarity between patches along the cycle.

Note that $\mathcal{T}$ is *only used in training* and is deliberately designed to be weak, so as to place the burden of representation on ϕ. At test time, the learned ϕ is used directly for computing correspondences. In the following, we first formalize cycle-consistent tracking loss functions and then describe our architecture for mid-level correspondence.

3.1. Cycle-Consistency Losses

We describe a formulation of cycle-consistent tracking and use it to succinctly express loss functions based on temporal cycle-consistency.

3.1.1 Recurrent Tracking Formulation

Consider as inputs a sequence of video frames $I_{t-k:t}$ and a patch p_t taken from I_t. These pixel inputs are mapped to a feature space by an encoder ϕ, such that $x_{t-k:t}^I = \phi(I_{t-k:t})$ and $x_t^p = \phi(p_t)$.

Let $\mathcal{T}$ be a differentiable operation $x_s^I \times x_t^p \mapsto x_s^p$, where s and t represent time steps. The role of $\mathcal{T}$ is to localize the patch features x_s^p in image features x_s^I that are most similar to x_t^p. We can apply $\mathcal{T}$ iteratively in a forward manner i times from $t-i$ to $t-1$:

$$\mathcal{T}^{(i)}(x_{t-i}^I, x^p) = \mathcal{T}(x_{t-1}^I, \mathcal{T}(x_{t-2}^I, ...\mathcal{T}(x_{t-i}^I, x^p)))$$

By convention, the tracker $\mathcal{T}$ can be applied backwards i times from time $t-1$ to $t-i$:

$$\mathcal{T}^{(-i)}(x_{t-1}^I, x^p) = \mathcal{T}(x_{t-i}^I, \mathcal{T}(x_{t-i+1}^I, ...\mathcal{T}(x_{t-1}^I, x^p)))$$

3.1.2 Learning Objectives

The following learning objectives rely on a measure of agreement $l_\theta(x_t^p, \hat{x}_t^p)$ between the initial patch and re-localized patch (defined in Section 3.2).

Tracking: The cycle-consistent loss $\mathcal{L}_{long}^i$ is defined as

$$\mathcal{L}_{long}^i = l_\theta(x_t^p, \mathcal{T}^{(i)}(x_{t-i+1}^I, \mathcal{T}^{(-i)}(x_{t-1}^I, x_t^p))).$$

The tracker attempts to follow features backward and then forward i steps in time to re-arrive to the initial query, as depicted in Figure 4a.

Skip Cycle: In addition to cycles through consecutive frames, we also allow skipping through time. We define the loss on a two-step skip-cycle as $\mathcal{L}_{skip}^i$:

$$\mathcal{L}_{skip}^i = l_\theta(x_t^p, \mathcal{T}(x_t^I, \mathcal{T}(x_{t-i}^I, x_t^p))).$$

This attempts longer-range matching by skipping to the frame i steps away.

Feature Similarity: We explicitly require the query patch x_t^p and localized patch $\mathcal{T}(x_{t-i}^I, x_t^p)$ to be similar in feature space. This loss amounts to the negative Frobenius inner product between spatial feature tensors:

$$\mathcal{L}_{sim}^i = -\langle x_t^p, \mathcal{T}(x_{t-i}^I, x_t^p) \rangle$$

In principle, this loss can further be formulated as the inlier loss from [54]. The overall learning objective sums over the k possible cycles, with weight $\lambda = 0.1$:

$$\mathcal{L} = \sum_{i=1}^{k} \mathcal{L}_{sim}^i + \lambda \mathcal{L}_{skip}^i + \lambda \mathcal{L}_{long}^i.$$

3.2. Architecture for Mid-level Correspondence

The learning objective thus described can be used to train arbitrary differentiable tracking models. In practice, the architecture of the encoder determines the type of correspondence captured by the acquired representation. In this work, we are interested in a model for mid-level temporal correspondence. Accordingly, we choose the representation to be a mid-level deep feature map, coarser than pixel space but with sufficient spatial resolution to support tasks that require localization. An overview is provided in Figure 4b.

3.2.1 Spatial Feature Encoder ϕ

We compute spatial features with a ResNet-50 architecture [18] without res_5 (the final 3 residual blocks). We reduce the spatial stride of res_4 for larger spatial outputs. Input frames are 240×240 pixels, randomly cropped from video frames re-scaled to have $\texttt{min}(H, W) = 256$. The size of the spatial feature of the frame is thus 30×30. Image patches are 80×80, randomly cropped from the full 240×240 frame, so that the feature is 10×10. We perform l_2 normalization on the channel dimension of spatial features to facilitate computing cosine similarity.

3.2.2 Differentiable Tracker $\mathcal{T}$

Given the representation from the encoder, we perform tracking with $\mathcal{T}$. As illustrated in Figure 4b, the differentiable tracker is composed of three main components.

Affinity function f provides a measure of similarity between coordinates of spatial features x^I and x^p. We denote the affinity function as $f(x^I, x^p) := A$, such that $f : \mathbb{R}^{c\times 30\times 30} \times \mathbb{R}^{c\times 10\times 10} \rightarrow \mathbb{R}^{900\times 100}$.

A generic choice for computing the affinity is the dot product between embeddings, referred to in recent literature as attention [67, 72] and more historically known as normalized cross-correlation [10, 35]. With spatial grid j in feature x^I as $x^I(j)$ and the grid i in x^p as $x^p(i)$,

$$A(j, i) = \frac{\exp\left(x^I(j)^\intercal x^p(i)\right)}{\sum_j \exp\left(x^I(j)^\intercal x^p(i)\right)} \tag{1}$$

where the similarity $A(j, i)$ is normalized by the softmax over the spatial dimension of x^I, for each $x^p(i)$. Note that the affinity function is defined for any feature dimension.

Localizer g takes affinity matrix A as input and estimates localization parameters θ corresponding to the patch in feature x^I which best matches x^p. g is composed of two convolutional layers and one linear layer. We restrict g to output 3 parameters for the bilinear sampling grid (i.e. simpler than [23]), corresponding to 2D translation and rotation: $g(A) := \theta$, where $g : \mathbb{R}^{900\times 100} \rightarrow \mathbb{R}^3$. The expressiveness of g is intentionally limited so as to place the burden of representation on the encoder (see Appendix B).

Bilinear Sampler h uses the image feature x^I and θ predicted by g to perform bilinear sampling to produce a new patch feature $h(x^I, \theta)$ which is in the same size as x^p, such that $h : \mathbb{R}^{c\times 30\times 30} \times \mathbb{R}^3 \rightarrow \mathbb{R}^{c\times 10\times 10}$.

3.2.3 End-to-end Joint Training

The composition of encoder ϕ and $\mathcal{T}$ forms a differentiable patch tracker, allowing for end-to-end training of ϕ and $\mathcal{T}$:

$$x^I, x^p = \phi(I), \phi(p)$$
$$\mathcal{T}(x^I, x^p) = h(x^I, g(f(x^I, x^p))).$$

Alignment Objective l_θ is applied in the cycle-consistent losses $\mathcal{L}^i_{long}$ and $\mathcal{L}^i_{skip}$, measuring the error in alignment between two patches. We follow the formulation introduced by [53]. Let $M(\theta_{x^p})$ correspond to the bilinear sampling grids used to form a patch feature x^p from image feature x^I. Assuming $M(\theta_{x^p})$ contains n sampling coordinates, the alignment objective is defined as:

$$l_\theta(x^p_*, \hat{x}^p_t) = \frac{1}{n}\sum_{i=1}^{n} ||M(\theta_{x^p_*})_i - M(\theta_{\hat{x}^p_t})_i||_2^2$$

4. Experiments

We report experimental results for a model trained on the VLOG dataset [12] from scratch; training on other large video datasets such as Kinetics gives similar results (see Appendix A.3). The trained representation is evaluated *without fine-tuning* on several challenging video propagation tasks: DAVIS-2017 [48], JHMDB [26] and Video Instance-level Parsing (VIP) [85]. Through various experiments, we show that the acquired representation generalizes to a range of visual correspondence tasks (see Figure 5).

4.1. Common Setup and Baselines

Training. We train the model on the VLOG dataset [12] without using any annotations or pre-training. The VLOG dataset contains 114K videos and the total length of the videos is 344 hours. During training, we set the number of past frames as $k = 4$. We train on a 4-GPU machine with a mini-batch size of 32 clips (8 clips per GPU), for 30 epochs. The model is optimized with Adam [32] with a learning rate of 0.0002 and momentum term $\beta_1 = 0.5, \beta_2 = 0.999$.

Inference. At test time, we use the trained encoder's representation to compute dense correspondences for video propagation. Given initial labels of the first frame, we propagate the labels to the rest of the frames in the video. Labels are given by specified targets for the first frame of each task, with instance segmentation masks for DAVIS-2017 [48], human pose keypoints JHMDB [26], and both instance-level and semantic-level masks for VIP [85]. The labels of each pixel are discretized to C classes. For segmentation masks, C is the number of instance or semantic labels. For keypoints, C is the number of keypoints. We

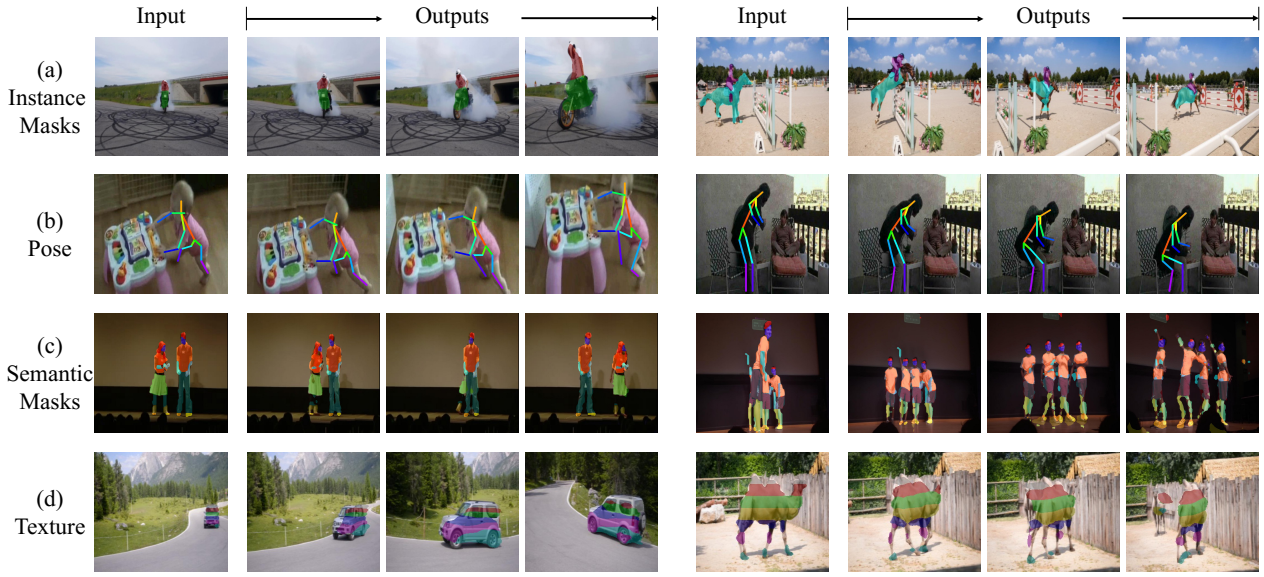


Figure 5: Visualizations of our propagation results. Given the labels as input in the first frame, our feature can propagate them to the rest of frames, without further fine-tuning. The labels include (a) instance masks in DAVIS-2017 [48], (b) pose keypoints in JHMDB [26], (c) semantic masks in VIP [85] and even (d) texture map.

include a background class. We propagate the labels in the feature space. The labels in the first frame are one-hot vectors, while propagated labels are soft distributions.

Propagation by k-NN. Given a frame I_t and a frame I_{t-1} with labels, we compute their affinity in feature space: $A_{t-1,t} = f(\phi(I_{t-1}), \phi(I_t))$ (Eq. 1). We compute label y_i of pixel i in I_t as

$$y_i = \sum_j A_{t-1,t}(j,i) y_j, \tag{2}$$

where $A_{t-1,t}(j,i)$ is the affinity between pixels i in I_t and j in I_{t-1}. We propagate from the top-5 pixels with the greatest affinity $A_{t-1,t}(j,i)$ for each pixel i. Labels are propagated from $I_{t-1:t-K}$, as well as I_1, and averaged. Finally, we up-sample the label maps to image size. For segmentation, we use the `argmax` of the class distribution of each pixel. For keypoints, we choose the pixel with the maximum score for each keypoint type.

Baselines. We compare with the following baselines:

- **Identity**: Always copy the first frame labels.
- **Optical Flow** (FlowNet2 [22]): A state-of-the-art method for predicting optical flow with neural networks [22]. We adopt the open-source implementation which is trained with synthetic data in a supervised manner. For a target frame I_t, we compute the optical flow from frame I_{t-1} to I_t and warp the labels in I_{t-1} to I_t.
- **SIFT Flow** [39]: For a target frame I_t, we compute the SIFT Flow between I_t and its previous frames. We propagate the labels in K frames before I_t and the first frame via SIFT Flow warping. The propagation results are averaged to compute the labels for I_t.
- **Transitive Invariance** [74]: A self-supervised approach that combines multiple objectives: (i) visual tracking on raw video [73] and (ii) spatial context reasoning [9]. We use the open-sourced pre-trained VGG-16 [58] model and adopt our proposed inference procedure.
- **DeepCluster** [8]: A self-supervised approach which uses a K-means objective to iteratively update targets and learn a mapping from images to targets. It is trained on the ImageNet dataset without using annotations. We apply the trained model with VGG-16 and adopt the same inference procedure as our method.
- **Video Colorization** [69]: A self-supervised approach for label propagation. Trained on the Kinetics [29] dataset, it uses color propagation as self-supervision. The architecture is based on 3D ResNet-18. We report their results.
- **ImageNet Pre-training** [18]: The conventional setup for supervised training of ResNet-50 on ImageNet.
- **Fully-Supervised Methods**: We report fully-supervised methods for reference, which not only use ImageNet pre-training but also fine-tuning on the target dataset. Note that these methods do not always follow the inference procedure used with method, and labels of the first frame are not used for JHMDB and VIP at test time.

4.2. Instance Propagation on DAVIS-2017

We apply our model to video object segmentation on the DAVIS-2017 validation set [48]. Given the initial masks of the first frame, we propagate the masks to the rest of the frames. Note that there can be multiple instances in the first frame. We follow the standard metrics including the region

model	Supervised	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)
Identity		22.1	23.6
Random Weights (ResNet-50)		12.4	12.5
Optical Flow (FlowNet2) [22]		26.7	25.2
SIFT Flow [39]		33.0	35.0
Transitive Inv. [74]		32.0	26.8
DeepCluster [8]		37.5	33.2
Video Colorization [69]		34.6	32.7
Ours (ResNet-18)		40.1	38.3
Ours (ResNet-50)		**41.9**	**39.4**
ImageNet (ResNet-50) [18]	✓	50.3	49.0
Fully Supervised [81, 7]	✓	55.1	62.1

Table 1: Evaluation on instance mask propagation on DAVIS-2017 [48]. We follow the standard metric on region similarity $\mathcal{J}$ and contour-based accuracy $\mathcal{F}$.

similarity $\mathcal{J}$ (IoU) and the contour-based accuracy $\mathcal{F}$. We set $K = 7$, the number of reference frames in the past.

We show comparisons in Table 1. Comparing to the recent Video Colorization approach [69], our method is 7.3% in $\mathcal{J}$ and 6.7% in $\mathcal{F}$. Note that although we are only 4.4% better than the DeepCluster baseline in $\mathcal{J}$, we are better in contour accuracy $\mathcal{F}$ by 6.2%. Thus, DeepCluster does not capture dense correspondence on the boundary as well.

For fair comparisons, we also implemented our method with a ResNet-18 encoder, which has less parameters compared to the VGG-16 in [74, 8] and the 3D convolutional ResNet-18 in [69]. We observe that results are only around 2% worse than our model with ResNet-50, which is still better than the baselines.

While the ImageNet pre-trained network performs better than our method on this task, we argue it is easy for the ImageNet pre-trained network to recognize objects under large variation as it benefits from curated object-centric annotation. Though our model is only trained on indoor scenes without labels, it generalizes to outdoor scenes.

Although video segmentation is an important application, it does not necessarily show that the representation captures dense correspondence.

4.3. Pose Keypoint Propagation on JHMDB

To see whether our method is learning more spatially precise correspondence, we apply our model on the task of keypoint propagation on the split 1 validation set of JHMDB [26]. Given the first frame with 15 labeled human keypoints, we propagate them through time. We follow the evaluation of the standard PCK metric [82], which measures the percentage of keypoints close to the ground truth in different thresholds of distance. We set the number of reference frames same as experiments in DAVIS-2017.

As shown in Table 2, our method outperforms all self-supervised baselines by a large margin. We observe that SIFT Flow actually performs better than other self-supervised learning methods in PCK@.1. Our method outperforms SIFT Flow by 8.7% in PCK@.1 and 9.9% in PCK@.2. Notably, our approach is only 0.7% worse than ImageNet pre-trained features in PCK@.1 and performs better in PCK@.2.

model	Supervised	PCK@.1	PCK@.2
Identity		43.1	64.5
Optical Flow (FlowNet2) [22]		45.2	62.9
SIFT Flow [39]		49.0	68.6
Transitive Inv. [74]		43.9	67.0
DeepCluster [8]		43.2	66.9
Video Colorization [69]		45.2	69.6
Ours (ResNet-18)		57.3	78.1
Ours (ResNet-50)		**57.7**	**78.5**
ImageNet (ResNet-50) [18]	✓	58.4	78.4
Fully Supervised [59]	✓	68.7	92.1

Table 2: Evaluation on pose propagation on JHMDB [26]. We report the PCK in different thresholds.

4.4. Semantic and Instance Propagation on VIP

We apply our approach on the Video Instance-level Parsing (VIP) dataset [85], which is densely labeled with semantic masks for different human parts (e.g., hair, right arm, left arm, coat). It also has instance labels that differentiate humans. Most interestingly, the duration of a video ranges from 10 seconds to 120 seconds in the dataset, which is much longer than aforementioned datasets.

We test our method on the validation set of two tasks in this dataset: (i) The first task is to propagate the semantic human part labels from the first frame to the rest of the video, and evaluate with the mean IoU metric; (ii) In the second task, the labels in the first frame are given with not only the semantic labels but also the instance identity. Thus, the model must differentiate the different arms of different human instances. We use the standard instance-level human parsing metric [37], mean Average Precision, for overlap thresholds varying from 0.1 to 0.9. Since part segments are relatively small (compared to objects in DAVIS-2017), we increase the input image size to 560×560 for inference, and use two reference frames, including the first frame.

Semantic Propagation. As shown with the mIoU metric in Table 3, our method again exceeds all self-supervised baselines by a large margin (a [69] model is currently not available). ImageNet pre-trained models have the advantage of semantic annotation and thus do not necessarily have to perform tracking. As shown in Figure 5(c), our method is able to handle occlusions and multiple instances.

Part Instance Propagation. This task is more challenging. We show the results in mean AP^r_{vol} in Table 3. Our method performs close to the level of ImageNet pre-trained features. We show different radial thresholds for average precision (AP^r_{vol}) in Table 4. ImageNet pre-trained features performs better under smaller thresholds and worse under larger thresholds, suggesting that it has an advantage in finding coarse correspondence while our method is more capable of spatial precision.

model	Supervised	mIoU	AP^r_{vol}
Identity		13.6	4.0
Optical Flow (FlowNet2) [22]		16.1	8.3
SIFT Flow [39]		21.3	10.5
Transitive Inv. [74]		19.4	5.0
DeepCluster [8]		21.8	8.1
Ours (ResNet-50)		**28.9**	**15.6**
ImageNet (ResNet-50) [18]	✓	34.7	16.1
Fully Supervised [85]	✓	37.9	24.1

Table 3: Evaluation on propagating human part labels in Video Instance-level Parsing (VIP) dataset [85]. We measure *Semantic Propagation* with mIoU and *Part Instance Propagation* in AP^r_{vol}.

model	AP^r_{vol}	IoU threshold 0.3	0.5	0.7
Ours (ResNet-50)	15.6	23.0	**12.7**	**5.4**
ImageNet (ResNet-50) [18]	**16.1**	**24.2**	11.9	4.8

Table 4: A more detailed analysis of different thresholds for *Part Instance Propagation* on the VIP dataset [85].

4.5. Texture Propagation

The acquired representation allows for propagation of not only instance and semantic labels, but also textures. We visualize texture propagation in Figure 5 (d); these videos are samples from DAVIS-2017 [48]. We "paint" a texture of 6 colored stripes on an the object in the first frame and propagate it to the rest of the frames using our representation. We observe that the structure of the texture is well preserved in the following frames, demonstrating that the representation allows for finding precise correspondence smoothly though time. See the project page for video examples.

4.6. Video Frame Reconstructions

Though we do not optimize for pixel-level objectives at training time, we can evaluate how well our method performs on pixel-level reconstruction. Specifically, given two images I_s and I_t distant in time in a video, we compute coordinate-wise correspondences under the acquired representation and generate a flow field for pixel movement between I_s and I_t. We then upsample the flow field to the same size as the image and warp it on Image I_s to generate a new image I'_t (as shown in Figure 7). We compare the L1 distance between I'_t and I_t in RGB space and report the reconstruction errors in Table 5.

For fair comparison, we perform this experiment on the DAVIS-2017 validation set, which none of the reported methods have seen. We experiment with two time gaps, 5 and 10 frames. For the smaller gap, FlowNet2 [22] performs reasonably well, whereas reconstruction degrades for larger gaps. In both cases, our method performs better than FlowNet2 and the ImageNet pre-trained network. This is encouraging: our method is not trained with pixel-level losses, yet out-performs methods trained with pixel-level tasks and human supervision.

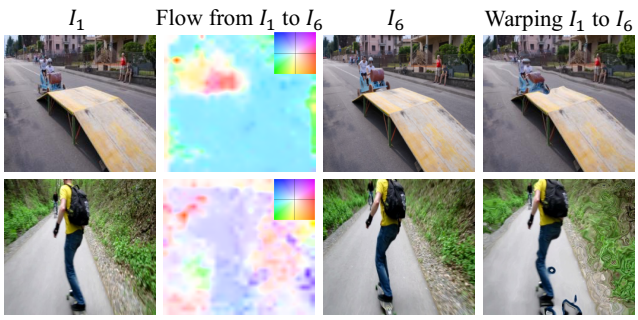


Figure 6: Given I_1, I_6 which have 5-frame gap, we compute the long-range flows between them with our representation. This flow can be used to warp I_1 to generate image similar to I_6.

model	5-F	10-F
Identity	82.0	97.7
Optical Flow (FlowNet2) [22]	62.4	90.3
ImageNet (ResNet-50) [18]	64.0	79.2
Ours (ResNet-50)	**60.4**	**76.4**

Table 5: We compute the long-range flow on two frames and warp the first one with the flow. We compare the warped frame with the second frame in L1 distance. The gaps are 5 or 10 frames.

5. Limitations and Future Work

While in principle our method should keep improving with more data, in practice, learning seems to plateau after a moderate amount of training (i.e. 30 epochs). An important next step is thus how to better scale to larger, noisier data. A crucial component is improving robustness to occlusions and partial observability, for instance, by using a better search strategy for finding cycles at training time. Another issue is deciding *what* to track at training time. Picking patches at random can result in issues such as stationary background patches and tracking ambiguity – e.g. how should one track a patch containing two objects that eventually diverge? Jointly learning what to track may also give rise to unsupervised object detection. Finally, incorporating more context for tracking both at training and test time may be important for learning more expressive models of spatial-temporal correspondence.

We hope this work is a step toward learning from the abundance of visual correspondence inherent in raw video in a scalable and end-to-end manner. While our experiments show promising results at certain levels of correspondence, much work remains to cover the full spectrum.

Acknowledgements: We thank members of the BAIR community for helpful discussions and feedback, and Sasha Sax and Michael Janner for draft comments. AJ is supported by the P.D. Soros Fellowship. XW is supported by the Facebook PhD Fellowship. This work was also supported, in part, by NSF grant IIS-1633310 and Berkeley DeepDrive.

References

[1] Recurrent Tracking using Multifold Consistency. *CVPR Workshop on PETS*, 2007. 3

[2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *ICCV*, 2015. 2

[3] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *CVPR*, 2008. 3

[4] A. Bansal, S. Ma, D. Ramanan, and Y. Sheikh. Recycle-gan: Unsupervised video retargeting. In *ECCV*, 2018. 3

[5] T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *CVPR*, 2009. 3

[6] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004. 3

[7] S. Caelles, K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *CVPR*, 2017. 7

[8] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018. 6, 7, 8

[9] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 6

[10] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. Van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. *arXiv*, 2015. 1, 3, 5

[11] D. A. Forsyth and J. Ponce. *Computer Vision - A Modern Approach, Second Edition.* Pitman, 2012. 3

[12] D. F. Fouhey, W. Kuo, A. A. Efros, and J. Malik. From lifestyle vlogs to everyday interactions. In *CVPR*, 2018. 5

[13] P. Fldik. Learning Invariance from Transformation Sequences. *Neural Computation*, 3(2):194–200, June 1991. 2

[14] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. 2017. 3

[15] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun. Unsupervised learning of spatiotemporally coherent metrics. *ICCV*, 2015. 2

[16] B. Ham, M. Cho, C. Schmid, and J. Ponce. Proposal flow. In *CVPR*, 2016. 3

[17] K. Han, R. S. Rezende, B. Ham, K.-Y. K. Wong, M. Cho, C. Schmid, and J. Ponce. Scnet: Learning semantic correspondence. *ICCV*, 2017. 3

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 6, 7, 8

[19] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *ECCV*, 2016. 1, 3

[20] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 1981. 3

[21] Q.-X. Huang and L. Guibas. Consistent shape maps via semidefinite programming. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing*, 2013. 3

[22] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017. 1, 3, 6, 7, 8

[23] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. *CoRR*, abs/1506.02025, 2015. 5

[24] J. Janai, F. Güney, A. Ranjan, M. Black, and A. Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *ECCV*, 2018. 3

[25] D. Jayaraman and K. Grauman. Learning image representations tied to egomotion. In *ICCV*, 2015. 2

[26] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *ICCV*, 2013. 5, 6, 7

[27] Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-backward error: Automatic detection of tracking failures. In *ICPR*, 2010. 3

[28] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *TPAMI*, 2012. 3

[29] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv:1705.06950*, 2017. 6, 12

[30] J. Kim, C. Liu, F. Sha, and K. Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *CVPR*, 2013. 3

[31] S. Kim, D. Min, B. Ham, S. Jeon, S. Lin, and K. Sohn. Fcss: Fully convolutional self-similarity for dense semantic correspondence. In *CVPR*, 2017. 3

[32] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv*, 2014. 5

[33] D. Larlus and A. Vedaldi. Anchornet: A weakly supervised network to learn geometry-sensitive features for semantic matching. In *CVPR*, 2017. 3

[34] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR*, 2011. 3

[35] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018. 3, 5

[36] N. Li and J. J. DiCarlo. Unsupervised natural experience rapidly alters invariant object representation in visual cortex. *Science (New York, N.Y.)*, 321(5895):1502–1507, Sept. 2008. 2

[37] Q. Li, A. Arnab, and P. H. Torr. Holistic, instance-level human parsing. *arXiv*, 2017. 7

[38] Y. Li, M. Paluri, J. M. Rehg, and P. Dollár. Unsupervised learning of edges. In *CVPR*, 2016. 2

[39] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *TPAMI*, 2011. 3, 6, 7, 8

[40] S. Liu, G. Zhong, S. De Mello, J. Gu, M.-H. Yang, and J. Kautz. Switchable temporal propagation network. *ECCV*, 2018. 2

[41] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *IJCAI*, 1981. 3

[42] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. 2017. 2

[43] D. Mahajan, F.-C. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur. Moving gradients: A path-based method for plausible image interpolation. In *Proceedings of SIGGRAPH, ACM Transactions on Graphics*, 2009. 3

[44] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv*, 2015. 2

[45] S. Meister, J. Hur, and S. Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. *AAAI*, 2018. 3

[46] E. Mémin and P. Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 1998. 3

[47] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. In *CVPR*, 2017. 2

[48] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 5, 6, 7, 8

[49] D. Ramanan, D. A. Forsyth, and A. Zisserman. Strike a pose: Tracking people by finding stylized poses. In *CVPR*, 2005. 3

[50] A. Ranjan and M. Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017. 1, 3

[51] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, 2015. 3

[52] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Deepmatching: Hierarchical deformable dense matching. *IJCV*, 2016. 3

[53] I. Rocco, R. Arandjelović, and J. Sivic. Convolutional neural network architecture for geometric matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 5

[54] I. Rocco, R. Arandjelovic, and J. Sivic. End-to-end weakly-supervised semantic alignment. In *CVPR*, 2018. 3, 4

[55] M. Rubinstein, C. Liu, and W. T. Freeman. Towards longer long-range motion trajectories. 2012. 3

[56] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. *ICCV*, 2008. 3

[57] I. K. Sethi and R. Jain. Finding Trajectories of Feature Points in a Monocular Image Sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(1):56–73, Jan. 1987. 3

[58] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014. 6

[59] J. Song, L. Wang, L. Van Gool, and O. Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. In *CVPR*, 2017. 7

[60] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using LSTMs. *arXiv*, 2015. 2

[61] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *CVPR*, 2010. 3

[62] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018. 1, 3

[63] H.-Y. Tung, H.-W. Tung, E. Yumer, and K. Fragkiadaki. Self-supervised learning of motion capture. In *NIPS*, 2017. 2

[64] N. Ufer and B. Ommer. Deep semantic feature matching. In *CVPR*, 2017. 3

[65] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, 2017. 1, 3

[66] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. 2

[67] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Neural Information Processing Systems (NIPS)*, 2017. 5

[68] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv*, 2017. 3

[69] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy. Tracking emerges by colorizing videos. 2017. 3, 6, 7

[70] G. Wallis. Spatio-temporal influences at the neural level of object recognition. *Network: Computation in Neural Systems*, 9(2):265–278, Jan. 1998. 2

[71] N. Wang and D.-Y. Yeung. Learning a deep compact image representation for visual tracking. In *NIPS*, 2013. 1, 3

[72] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018. 5

[73] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 2, 6

[74] X. Wang, K. He, and A. Gupta. Transitive invariance for self-supervised visual representation learning. In *ICCV*, 2017. 2, 6, 7, 8

[75] Y. Wang, Y. Yang, and W. Xu. Occlusion aware unsupervised learning of optical flow. 2018. 3

[76] L. Wiskott and T. J. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, Apr. 2002. 2

[77] J. N. Wood. A smoothness constraint on the development of object recognition. *Cognition*, 153:140–145, 2016. 2

[78] J. N. Wood and S. M. W. Wood. The development of newborn object recognition in fast and slow visual worlds. *Proceedings. Biological Sciences*, 283(1829), Apr. 2016. 2

[79] H. Wu, A. C. Sankaranarayanan, and R. Chellappa. In Situ Evaluation of Tracking Algorithms Using Time Reversed Chains. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007. 3

[80] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 3

[81] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos. Efficient video object segmentation via network modulation. 2018. 7

[82] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *TPAMI*, 2013. 7

[83] D.-J. Yi, N. B. Turk-Browne, J. I. Flombaum, M.-S. Kim, B. J. Scholl, and M. M. Chun. Spatiotemporal object continuity in human ventral visual cortex. *Proceedings of the National Academy of Sciences*, 105(26):8840–8845, July 2008. 2

[84] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018. 3

[85] Q. Zhou, X. Liang, K. Gong, and L. Lin. Adaptive temporal encoding network for video instance-level human parsing. In *ACM MM*, 2018. 5, 6, 7, 8

[86] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 3

[87] T. Zhou, Y. Jae Lee, S. X. Yu, and A. A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *CVPR*, 2015. 3

[88] T. Zhou, P. Krahenbuhl, M. Aubry, Q. Huang, and A. A. Efros. Learning dense correspondence via 3d-guided cycle consistency. In *CVPR*, 2016. 2, 3

[89] X. Zhou, M. Zhu, and K. Daniilidis. Multi-image matching via fast alternating minimization. In *ICCV*, 2015. 3

[90] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. 2017. 3

[91] W. Zou, S. Zhu, K. Yu, and A. Y. Ng. Deep Learning of Invariant Features via Simulated Fixations in Video. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 3203–3211. Curran Associates, Inc., 2012. 2

Appendix A. Ablations

A.1. Removing Skip-Cycles

Removing the skip-cycle loss – i.e. keeping only the long tracking cycle loss and dense similarity loss – results in worse performance when applying the representations to the DAVIS-2017 dataset. This suggests the skip-cycle loss is useful in cases of occlusion or drift, and provides supplementary training data (c.f. Table 6).

Experiment	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)
Ours	41.9	39.4
Ours without Skip Cycles	39.5	37.9

Table 6: Removing Skip Cycles, test on DAVIS.

A.2. Effect of k in k-NN Label Propagation

We vary the number of nearest neighbors used in voting for label propagation (Eq. 2), finding that aggregating fewer nearest neighbors improves performance (c.f. Table 7).

Experiment	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)
Ours (5-NN)	41.9	39.4
Ours (20-NN)	40.8	38.5
Ours (10-NN)	41.5	39.1
Ours (1-NN)	41.0	38.9

Table 7: Effect of k in k-NN Label Propagation, test on DAVIS.

A.3. Training with the Kinetics Dataset

Besides the VLOG dataset, we have also trained our model on the Kinetics Dataset [29], which contains around 230K training videos (with 10s per video). Compared to the VLOG dataset, the Kinetics dataset contains more videos under less environment constraints: There are videos with both indoor and outdoor scenes; some videos also have large camera motion. After applying the learned representation for label propagation on DAVIS, we observe similar performance by training with VLOG and Kinetics datasets (c.f. Table 8).

Experiment	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)
Ours (VLOG)	41.9	39.4
Ours (Kinetics)	42.5	39.2

Table 8: Train with VLOG / Kinetics, test on DAVIS.

A.4. Fine-tuning on the Test Domain

We emphasize that our method learns features that generalize even *without* fine-tuning. Here we study the effect of fine-tuning on the DAVIS training set before testing. We find this does not improve test set performance significantly (c.f. Table 9). There is a risk of overfitting since datasets like DAVIS are so small; this is part of the reason why unsupervised methods are desirable.

Experiment	$\mathcal{J}$(Mean)	$\mathcal{F}$(Mean)
Ours (ResNet-50)	41.9	39.4
Fine-tune	42.0	39.1

Table 9: Finetuning on DAVIS train before test.

Appendix B. Capacity of $\mathcal{T}$

As mentioned in Section 3.2, the tracking operation T is deliberately constrained in capacity in order to maximize the representational responsibility of ϕ. In our implementation, the only parameters learned by $\mathcal{T}$ are those of the localizer g, which processes the affinity tensor A to estimate the localization parameters. The affinity A ($\mathbb{R}^{900\times100}$) is first reshaped to a tensor with dimension $\mathbb{R}^{900\times10\times10}$ as the input for g. The localizer g is a small ConvNet with two convolutional layers (3×3 kernels with 512 channels) and one fully connected layer. The output of the ConvNet is a 3-dimension vector corresponding to 2D translation and rotation.

Appendix C. Correspondence Visualization

In Fig. 7 we visualize the correspondences (top-1 nearest neighbor) between regions with large movement in consecutive frames, comparing our features to ImageNet pre-trained features. Our method produces more detailed correspondence. However, for certain object-level tasks (e.g. DAVIS), high-level semantics (captured by ImageNet) are more useful than good correspondences, which explains the difference in performance.

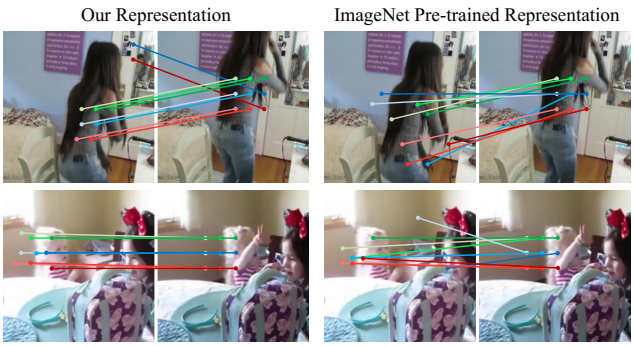


Figure 7: Visualizations of correspondence.