The 2017 DAVIS Challenge on Video Object Segmentation - CVPR 2017 Workshops

Lucid Data Dreaming for Object Tracking

Anna Khoreva[1] Rodrigo Benenson[2] Eddy Ilg[3] Thomas Brox[3] Bernt Schiele[1]

[1]Max Planck Institute for Informatics [2]Google [3]University of Freiburg

Abstract

Convolutional networks reach top quality in pixel-level object tracking but require a large amount of training data ($1k \sim 10k$) to deliver such results. We propose a new training strategy which achieves state-of-the-art results across three evaluation datasets while using $20\times \sim 100\times$ less annotated data than competing methods. Our approach is suitable for both for single and multiple object tracking.

Instead of using large training sets hoping to generalize across domains, we generate in-domain training data using the provided annotation on the first frame of each video to synthesize ("lucid dream"[1]) plausible future video frames. In-domain per-video training data allows us to train high quality appearance- and motion-based models, as well as tune the post-processing stage. This approach allows to reach competitive results even when training from only a single annotated frame, without ImageNet pre-training. Our results indicate that using a larger training set is not automatically better, and that for the tracking task a smaller training set that is closer to the target domain is more effective. This changes the mindset regarding how many training samples and general "objectness" knowledge are required for the object tracking task.

1. Introduction

In the last years the field of object tracking in videos has transitioned from bounding box [5] to pixel-level tracking [11, 17, 15]. Given a first frame labelled with the object masks, one aims to find the corresponding object pixels in future frames. Tracking objects at the pixel level enables a finer understanding of videos and is helpful for tasks such as video editing, rotoscoping, and summarisation.

Top performing results are currently obtained using convolutional networks (convnets) [8, 1, 9, 5]. Like most deep learning techniques, convnets for pixel-level object tracking benefit from large amounts of training data. Current state-of-the-art methods rely, for instance, on pixel accurate foreground/background annotations of $\sim 2k$ video frames [8, 1] or $\sim 10k$ images [9]. Labelling videos at the pixel level is a laborious task (compared e.g. to drawing bounding boxes for detection), and creating a large training set requires significant annotation effort.

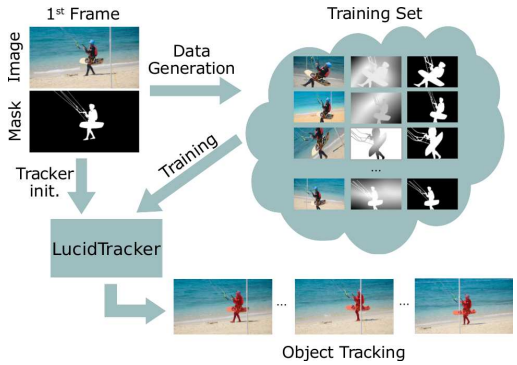


Figure 1: Starting from scarce annotations we synthesize in-domain data to train a specialized pixel-level object tracker.

In this work we aim to reduce the necessity for such large volumes of training data. It is traditionally assumed that convnets requires large training sets to perform best. We show that for video object tracking having a larger training set is not automatically better and that improved results can be obtained by using $20\times \sim 100\times$ less training data than previous approaches [1, 9]. The main insight of our work is that for pixel-level object tracking using few training frames ($1 \sim 100$) in the target domain is more useful than using large training volumes across domains ($1k \sim 10k$).

To ensure a sufficient amount of training data close to the target domain, we develop a new technique for synthesizing training data tailored for the object tracking scenario. We call this data generation strategy "*lucid dreaming*", where the first frame and its annotation mask are used to generate plausible future video frames. The goal is to produce a large training set of reasonably realistic images which capture the expected appearance variations in future video frames, and thus is, by design, close to the target domain.

Our approach is suitable for both for single and multiple object tracking. Enabled by the proposed data generation strategy and the efficient use of optical flow, we are able to achieve high quality results while using only ~ 100 individual annotated training frames. Moreover, in the extreme case with only a single annotated frame (zero pre-training), we still obtain competitive tracking results.

2. Related work

Pixel-level tracking. In this paper we focus on generating a foreground versus background pixel-wise object labelling

[1]In a lucid dream the sleeper is aware that he or she is dreaming and is sometimes able to control the course of the dream.

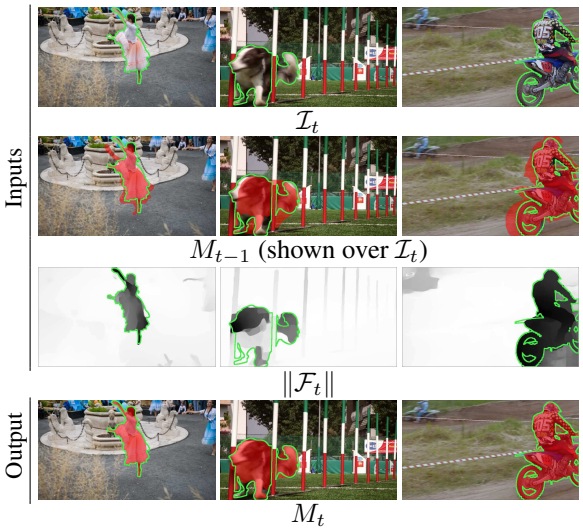


Figure 2: Data flow examples. $\mathcal{I}_t$, $\mathcal{F}_t$, M_{t-1} are the inputs, M_t is the resulting output.

for video starting from a first manually annotated frame. Multiple strategies have been proposed to solve this task.

Mask propagation: Appearance similarity and motion smoothness across time is used to propagate the first frame annotation across the video [12, 21].

Video saliency: These methods extract the main foreground object pixel-level space-time tube [20, 7].

Convnets: Recently convnets have been proposed for pixel-level tracking. [1] trains a generic object saliency network, and fine-tunes it per-video to make the output sensitive to the specific object instance being tracked. [9] uses a similar strategy, but also feeds the mask from the previous frame as guidance for the saliency network. Finally [8] mixes convnets with ideas of bilateral filtering.

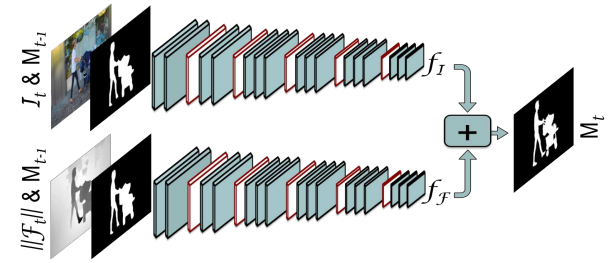
Our network architecture is similar to [1, 9]. However, we use a different strategy for training: [1, 8] rely on video training frames and [9] uses an external saliency dataset, while our approach focuses on using the first frame annotations provided with each targeted video benchmark without relying on external annotations.

Synthetic data. Like in our approach, previous works have also explored synthesizing training data. Synthetic renderings [13] and video games [18] have shown promise, but require 3d models. Compositing real images provides more realistic results, e.g. for text localization [4]. The closest work to ours is [14], which also generates video-specific training data using the first frame annotations. They use human skeleton annotations to improve pose estimation, while we employ mask annotations to improve object tracking.

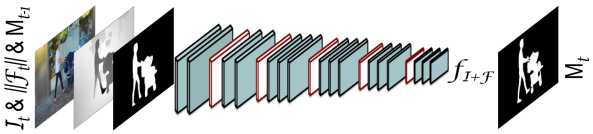
3. LucidTracker

3.1. Architecture

Approach. We model the pixel-level object tracking problem as a mask refinement task (mask: binary foreground/background labelling of the image) based on appearance



(a) Two stream architecture, where image $\mathcal{I}_t$ and optical flow information $\|\mathcal{F}_t\|$ are used to update mask M_{t-1} into M_t. See equation 1.



(b) One stream architecture, where 5 input channels: image $\mathcal{I}_t$, optical flow information $\|\mathcal{F}_t\|$ and mask M_{t-1} are used to estimate mask M_t.

Figure 3: Overview of the proposed one stream and two stream architectures. See §3.1.

and motion cues. From frame $t-1$ to frame t the estimated mask M_{t-1} is propagated to frame t, and the new mask M_t is computed as a function of the previous mask, the new image $\mathcal{I}_t$, and the optical flow $\mathcal{F}_t$, i.e. $M_t = f(\mathcal{I}_t, \mathcal{F}_t, M_{t-1})$. Since objects have a tendency to move smoothly through space in time, there are little changes from frame to frame and mask M_{t-1} can be seen as a rough estimate of M_t. Thus we require our trained convnet to learn to refine rough masks into accurate masks. Fusing the complementary image $\mathcal{I}_t$ and motion $\mathcal{F}_t$ cues exploits the information inherent to video and enables the model to segment well both static and moving objects.

Note that this approach is incremental, does a single forward pass over the video, and keeps no explicit model of the object appearance at frame t. We also consider adapting the model f per video, using the annotated first frame $\mathcal{I}_0$, M_0.

First frame. In the video object tracking task the mask for the first frame M_0 is given. This is the standard protocol of the benchmarks considered in Section 5.

RGB image $\mathcal{I}$. Typically a semantic labeller generates pixel-wise labels based on the input image (e.g. $M = g(\mathcal{I})$). We use an augmented semantic labeller with an input layer modified to accept 4 channels (RGB + previous mask) so as to generate outputs based on the previous mask estimate, e.g. $M_t = f_{\mathcal{I}}(\mathcal{I}_t, M_{t-1})$. Our approach is general and can leverage any existing semantic labelling architecture. We select the DeepLabv2 architecture with VGG base network [2], which is comparable to [8, 1, 9]; FusionSeg [7] uses ResNet.

Optical flow $\mathcal{F}$. We use flow in two complementary ways. First, to obtain a better estimate of M_t we warp M_{t-1} using the flow $\mathcal{F}_t$: $M_t = f_{\mathcal{I}}(\mathcal{I}_t, w(M_{t-1}, \mathcal{F}_t))$. Second, we use flow as a direct source of information about the mask M_t. As can be seen in Figure 2, when the object is moving

relative to background, the flow magnitude $\|\mathcal{F}_t\|$ provides a very reasonable estimate of the mask M_t. We thus consider using convnet specifically for mask estimation from flow: $M_t = f_{\mathcal{F}}(\mathcal{F}_t, w(M_{t-1}, \mathcal{F}_t))$, and merge it with the image-only version by naive averaging

$$M_t = 0.5 \cdot f_{\mathcal{I}}(\mathcal{I}_t, \ldots) + 0.5 \cdot f_{\mathcal{F}}(\mathcal{F}_t, \ldots). \quad (1)$$

We use the state-of-the-art optical flow method FlowNet2.0 [6], which itself is a convnet that computes $\mathcal{F}_t = h(\mathcal{I}_{t-1}, \mathcal{I}_t)$.

In our experiments $f_{\mathcal{I}}$ and $f_{\mathcal{F}}$ are trained independently. Our two stream architecture is illustrated in Figure 3a. We also explored expanding our network to accept 5 input channels (RGB + previous mask + flow magnitude) in one stream: $M_t = f_{\mathcal{I}+\mathcal{F}}(\mathcal{I}_t, \mathcal{F}_t, w(M_{t-1}, \mathcal{F}_t))$, but did not observe much difference in the performance compared to naive averaging. Our one stream architecture is illustrated in Figure 3b.

Multiple objects. The proposed framework can be extended to multiple object tracking. Instead of one additional channel for the previous frame mask we provide masks for each object in a separate channel, expanding the network to accept $3 + N$ input channels (RGB + N object masks): $M_t = f_{\mathcal{I}}\left(\mathcal{I}_t, w(M^1_{t-1}, \mathcal{F}_t), ..., w(M^N_{t-1}, \mathcal{F}_t)\right)$, where N is the number of objects.

For multiple object tracking task we employ one-stream architecture for the experiments and also explore using optical flow $\mathcal{F}$ and semantic segmentation $\mathcal{S}$ as additional input channels: $M_t{=}f_{\mathcal{I}+\mathcal{F}+\mathcal{S}}\left(\mathcal{I}_t, \mathcal{F}_t, \mathcal{S}_t, w(M^1_{t-1}, \mathcal{F}_t), ..., w(M^N_{t-1}, \mathcal{F}_t)\right)$. This allows to leverage the appearance model with semantic priors and motion information.

We use the state-of-the-art semantic segmentation method PSPNet [23], which itself is a convnet that computes $\mathcal{S}_t = h(\mathcal{I}_t)$.

We additionally experiment with ensembles of different variants, that allows to make the system more robust to the challenges inherent in videos. For our main results for multiple object tracking task we consider the ensemble of four models: $M_t{=}0.25 \cdot f_{\mathcal{I}+\mathcal{F}+\mathcal{S}} + 0.25 \cdot f_{\mathcal{I}+\mathcal{F}} + 0.25 \cdot f_{\mathcal{I}+\mathcal{S}} + 0.25 \cdot f_{\mathcal{I}}$, where we merge the outputs of the models by naive averaging. See Section 6 for more details.

Post-processing. As a final stage of our pipeline, we refine the generated mask M_t using DenseCRF [10] per frame. This adjusts small image details that the network might not have captured. It is known by practitioners that DenseCRF is quite sensitive to its parameters and can easily worsen results. We will use our lucid dreams to handle per-dataset CRF-tuning too, see Section 3.2.

We refer to our full system as `LucidTracker`, and as `LucidTracker`$^-$ when no post-processing is used.

3.2. Training modalities

Multiple modalities are available to train a tracker. **Training-free** approaches (e.g. BVS [12]) are fully hand-crafted systems with hand-tuned parameters, and thus do not require training data. They can be used as-is over different datasets. Supervised methods can also be trained to generate a **dataset-agnostic** model that can be applied over different datasets. Instead of using a fixed model for all cases, it is also possible to obtain specialized **per-dataset** models, either via self-supervision [22] or by using the first frame annotation of each video in the dataset as training/tuning set. Finally, inspired by traditional tracking techniques, we also consider adapting the model weights to the specific video at hand, thus obtaining **per-video** models. Section 5 reports results over these four training modalities (training-free, dataset-agnostic, per-dataset, and per-video).

Our `LucidTracker` obtains best results when first pre-trained on ImageNet, then trained per-dataset using all data from first frame annotations together, and finally fine-tuned per-video for each evaluated sequence.

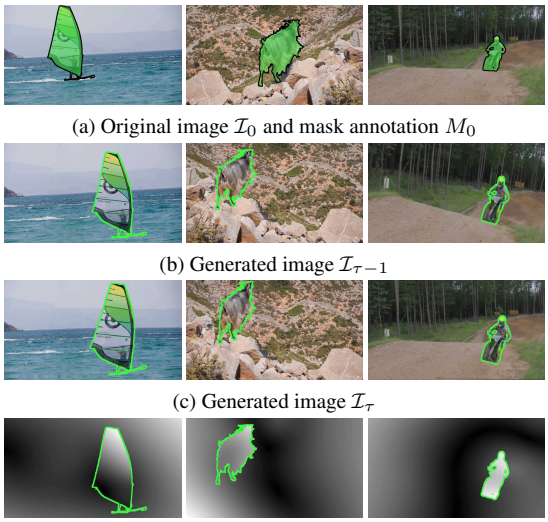
Training details. Models using pre-training are initialized with weights trained for image classification on ImageNet [19]. We then train per-dataset for 40k iterations. Models without ImageNet pre-training are initialized using the "Xavier" strategy [3]. The per-dataset training needs to be longer, using 100k iterations. For per-video fine-tuning 2k iterations are used for $f_{\mathcal{I}}$.

4. Lucid data dreaming

To train the function f one would think of using ground truth data for M_{t-1} and M_t (like [1]), however such data is expensive to annotate. [1] thus trains on a set of 30 videos ($\sim 2k$ frames) and requires the model to transfer across multiple tests sets. [9] side-steps the need for consecutive frames by generating synthetic masks M_{t-1} from a large saliency dataset of $\sim 10k$ images with their corresponding mask M_t. We propose a new data generation strategy to reach better results using only ~ 100 training frames.

Ideally training data should be as similar as possible to the test data, even subtle differences may affect quality. To ensure our training data is in-domain, we propose to generate it by synthesizing samples from the provided annotated first frame in each target video. This is akin to "lucid dreaming" as we intentionally "dream" the desired data, by creating images that are plausible hypothetical future frames of the video. The outcome of this process is a large set ($2.5k$ images) of frame pairs in the target domain with known optical flow and mask annotations, see Figure 4.

Synthesis process. The target domain for a tracker is the set of future frames of the given video. Traditional data augmentation via small image perturbation is insufficient



(a) Original image $\mathcal{I}_0$ and mask annotation M_0

(b) Generated image $\mathcal{I}_{\tau-1}$

(c) Generated image $\mathcal{I}_\tau$

(d) Generated flow magnitude $\|\mathcal{F}_\tau\|$

Figure 4: Lucid data dreaming examples.

to cover the expect variations across time, thus a task specific strategy is needed. Across the video the tracked object might change in illumination, deform, translate, be occluded, show different point of views, and evolve on top of a dynamic background. All of these aspects need to be captured when synthesizing future frames. We achieve this by cutting-out the foreground object, in-painting the background, perturbing both foreground and background, and finally recomposing the scene. This process is applied twice with randomly sampled transform parameters, resulting in a pair of frames $(\mathcal{I}_{\tau-1}, \mathcal{I}_\tau)$ with ground-truth pixel-level mask annotations $(M_{\tau-1}, M_\tau)$, optical flow $\mathcal{F}_\tau$, and occlusion regions, as the undergoing transformations are known. The object position in $\mathcal{I}_\tau$ is uniformly sampled, but the changes between $\mathcal{I}_{\tau-1}$, $\mathcal{I}_\tau$ are kept small to mimic the usual evolution between consecutive frames.

In more details, starting from an annotated image:

1. Illumination changes: we globally modify the image by randomly altering saturation S and value V (from HSV colour space) via $x' = a \cdot x^b + c$, where $a \in 1 \pm 0.05$, $b \in 1 \pm 0.3$, and $c \in \pm 0.07$.

2. Fg/Bg split: the foreground object is removed from the image $\mathcal{I}_0$ and a background image is created by inpainting the cut-out area.

3. Object motion: we simulate motion and shape deformations by applying global translation as well as affine and non-rigid deformations to the foreground object. For $\mathcal{I}_{\tau-1}$ the object is placed at any location within the image with a uniform distribution, and in $\mathcal{I}_\tau$ with a translation of $\pm 10\%$ of the object size relative to $\tau - 1$. In both frames we apply random rotation $\pm 30°$, scaling $\pm 15\%$ and thin-plate splines deformations of $\pm 10\%$ of the object size.

4. Camera motion: We additionally transform the background using affine deformations to simulate camera view changes. We apply here random translation, rotation, and scaling within the same ranges as for the foreground object.

5. Fg/Bg merge: finally $(\mathcal{I}_{\tau-1}, \mathcal{I}_\tau)$ are composed by blending the perturbed foreground with the perturbed background using Poisson matting. Using the known transformation parameters we also synthesize ground-truth pixel-level mask annotations $(M_{\tau-1}, M_\tau)$ and optical flow $\mathcal{F}_\tau$. Figure 4 shows example results. Albeit our approach does not capture appearance changes due to point of view, nor shadows, we see that already this rough modelling is effective to train our tracking models.

The same data synthesis strategy can be employed for multiple object tracking. Instead of manipulating one object we handle multiple objects at the same time, applying different transformations to each of them. In addition we also model partial and full occlusions between objects, mimicking plausible interactions of objects in the future frames.

Method	# training images	Flow $\mathcal{F}$	Dataset, mIoU DAVIS	YoutbObjs	SegTrck$_{v2}$
MP-Net [20]	~22.5k	✓	69.7	-	-
FusionSeg [7]	~95k	✓	71.5	67.9	-
BVS [12]	0	✗	66.5	59.7	58.4
ObjFlow [21]	0	✓	71.1	70.1	67.5
VPN [8]	~2.3k	✗	*75.0*	-	-
OSVOS [1]	~2.3k	✗	*79.8*	72.5	65.4
MaskTrack [9]	~11k	✓	80.3	72.6	70.3
LucidTracker	**24~126**	✓	**84.8**	**76.2**	**77.6**

Table 1: Results across three datasets. See §5.2.

5. Single object tracking

We present here results for single object tracking task: given a first frame labelled with the object mask, the goal is to find the corresponding object pixels in future frames.

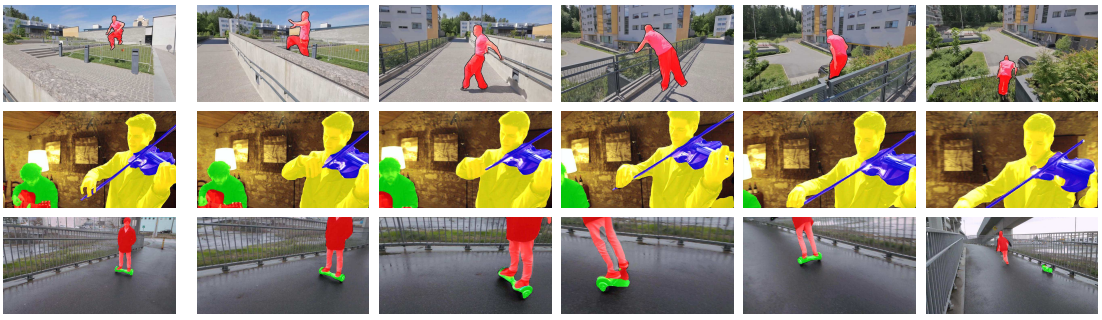
5.1. Experimental setup

We evaluate our method on three video object segmentation datasets: DAVIS [15], YouTubeObjects [17], and SegTrack$_{v2}$ [11]. These datasets provide diverse challenges with a mix of HD and low-res web videos, single or multiple salient objects per video, videos with flocks of similar looking instances, as well as the usual tracking challenges.

To measure the accuracy we use the mean intersection-over-union overlap (mIoU) between the ground truth and the predicted segmentation, averaged across all sequences.

5.2. Key results

Table 1 presents our main result and compares it to previous work. Our full system, `LucidTracker`, provides the best tracking quality across three datasets while being trained on each dataset using only one frame per video (50 frames for DAVIS, 126 for YouTubeObjects, 24 for SegTrack$_{v2}$), which is $20\times \sim 100\times$ less than the top competing methods. Ours is the first method to reach $>$ 75 mIoU on all three datasets.



1st frame, GT segment 20% 40% 60% 80% 100%

Figure 5: LucidTracker results. Frames sampled along the video duration (e.g. 50%: video middle point).

Variant	ImgNet pre-train.	per-dataset training	per-video fine-tun.	Dataset, mIoU DAVIS	YoutbObjs	SegTrck$_{v2}$
LucidTracker$^-$	✓	✓	✓	**83.7**	**76.2**	**76.8**
(no ImgNet)	✗	✓	✓	82.0	74.3	71.2
No per-video	✓	✓	✗	82.7	72.3	71.9
tuning	✗	✓	✗	78.4	69.7	68.2
Only per-	✓	✗	✓	79.4	-	70.4
-video tuning	✗	✗	✓	*80.5*	-	*66.8*

Table 2: Ablation study. Even with one frame annotation for only per-video tuning we obtain good results. See §5.3.

Compared to flow propagation methods such as BVS, ObjFlow, we obtain better results as we build per-video a stronger appearance model of the tracked object (embodied in the fine-tuned model). Compared to convnet learning methods such as VPN, OSVOS, MaskTrack, we require significantly less training data, yet obtain better results.

Conclusion. We show that less training data does not necessarily lead to poorer results and report the best known results for this task while using 24∼126 training frames.

5.3. Ablation study

Table 2 compares the effect of different ingredients in the `LucidTracker`$^-$ training. Results are obtained using RGB and flow, with warping, no CRF; $M_t = f\,(\mathcal{I}_t, w(M_{t-1}, \mathcal{F}_t))$.

We see that ImageNet pre-training does provide $2 \sim 5$ percent point improvement (e.g. $82.0 \rightarrow 83.7$ mIoU on DAVIS). Per-video fine-tuning (after doing per-dataset training) provides an additional $1 \sim 2$ percent point gain (e.g. 82.7→83.7 mIoU on DAVIS).

In the bottom row ("only per-video tuning"), the model is trained per-video without ImageNet pre-training nor per-dataset training, i.e. using a *single annotated training frame*. Even with such minimal amount of training data, we still obtain a surprisingly good performance (compare 80.5 on DAVIS to others in Table 1). This shows how effective is, by itself, the proposed training strategy based on lucid dreaming of the data.

Conclusion. Both ImageNet pre-training and per-video tuning of the models provide complementary gains over the default per-dataset training. Per-video training by itself, despite using a single annotated frame, provides already much of the needed information for the tracking task.

Method	test-dev set global mean
voigtlaender (5)	56.5
lalalafine123 (4)	57.4
wangzhe (3)	57.7
lixx (2)	66.1
LucidTracker (1)	**66.6**

(a) Results on test-dev set.

Method	test-challenge set global mean
voigtlaender (5)	57.7
haamooon (4)	61.5
vantam299 (3)	63.8
LucidTracker (2)	67.8
lixx (1)	**69.9**

(b) Results on test-challenge set.

Table 3: DAVIS 2017 challenge results.

6. Multiple object tracking

We present here an empirical evaluation of LucidTracker for multiple object tracking task: given a first frame labelled with the masks of several object instances, one aims is to find the corresponding masks of objects in future frames.

6.1. Experimental setup

For multiple object tracking we experiment on DAVIS 2017 [16]. This is a larger, more challenging dataset, where the video sequences have multiple objects in the scene. We evaluate our method on two test sets, the test-dev and test-challenge sets, each consists of 30 new videos.

To measure the accuracy of multiple object tracking we use the region (J) and boundary (F) measures [16]. As an overall measure the average of the J and F measures over all object instances is used.

6.2. Key results

Tables 3a and 3b presents the results of the 2017 DAVIS Challenge on test-dev and test-challenge sets [16].

Our main results are obtained via an ensemble of four different models. All models are initialized with weights trained for image classification on ImageNet and then tuned per-video. `LucidTracker` provides the best tracking quality on the test-dev set and shows competitive performance on the test-challenge set. The full system is trained using only one annotated frame per video, 30 frames overall.

Conclusion. We show that top results for multiple object tracking can be achieved using only the available annotation of the first frame for training.

Variant	$\mathcal{I}$	$\mathcal{F}$	$\mathcal{S}$	ensemble	CRF tuning	DAVIS 2017 test-dev		
						global mean	mIoU	mF
LucidTracker (ensemble)	✓	✓	✓	✓	✓	**66.6**	**63.4**	**69.9**
	✓	✓	✓	✓	✗	65.2	61.5	69.0
	✓	✓	✗	✗	✗	64.2	60.1	68.3
$\mathcal{I}+\mathcal{F}+\mathcal{S}$	✓	✓	✓	✗	✗	62.0	57.7	62.2
$\mathcal{I}+\mathcal{F}$	✓	✓	✗	✗	✗	61.3	56.8	65.8
$\mathcal{I}+\mathcal{S}$	✓	✗	✓	✗	✗	61.1	56.9	65.3
$\mathcal{I}$	✓	✗	✗	✗	✗	59.8	63.1	63.9

Table 4: Ablation study. DAVIS 2017, test-dev set.

6.3. Ablation study

In Table 4 we explore in more details how the different ingredients contribute to our results.

We see that adding extra channels to the system, either optical flow magnitude or semantic segmentation, or both provides $1\sim2$ percent point improvement.

Combining in ensemble four different models ($f_{\mathcal{I}+\mathcal{F}+\mathcal{S}} + f_{\mathcal{I}+\mathcal{F}} + f_{\mathcal{I}+\mathcal{S}} + f_{\mathcal{I}}$) enhances the results, bringing 3 percent point gain. CRF-tuning allows to further improve the results ($65.2\rightarrow66.6$ mIoU).

Conclusion. The results show that both flow and semantic priors provide a complementary signal to RGB image only. Despite its simplicity our ensemble strategy provides additional gain and leads to competitive results.

7. Conclusion

We have described a new convnet-based approach for pixel-level object tracking in videos. In contrast to previous work, we show that top results for single and multiple object tracking can be achieved without requiring external training datasets (neither annotated images nor videos). Our experiments indicate that it is not always beneficial to use additional training data, synthesizing training samples close to the test domain is more effective than adding more training samples from related domains.

Showing that training a convnet for object tracking can be done with only few (~ 100) training samples changes the mindset regarding how much general "objectness" knowledge is required to approach this problem [9, 7], and more broadly how much training data is required to train large convnets depending on the task at hand.

We hope these new results will fuel the ongoing evolution of convnet techniques for pixel-level object tracking.

References

[1] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixe, D. Cremers, and L. V. Gool. One-shot video object segmentation. In *CVPR*, 2017. 1, 2, 3, 4

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv:1606.00915*, 2016. 2

[3] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010. 3

[4] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, 2016. 2

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

[6] 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. 3

[7] S. D. Jain, B. Xiong, and K. Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmention of generic objects in videos. *arXiv:1701.05384*, 2017. 2, 4, 6

[8] V. Jampani, R. Gadde, and P. V. Gehler. Video propagation networks. *arXiv:1612.05478*, 2016. 1, 2, 4

[9] A. Khoreva, F. Perazzi, R. Benenson, B. Schiele, and A. Sorkine-Hornung. Learning video object segmentation from static images. In *arXiv:1612.02646*, 2016. 1, 2, 3, 4, 6

[10] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*. 2011. 3

[11] F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, 2013. 1, 4

[12] N. Maerki, F. Perazzi, O. Wang, and A. Sorkine-Hornung. Bilateral space video segmentation. In *CVPR*, 2016. 2, 3, 4

[13] N.Mayer, E.Ilg, P.Häusser, P.Fischer, D.Cremers, A.Dosovitskiy, and T.Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2

[14] D. Park and D. Ramanan. Articulated pose estimation with tiny synthetic videos. In *CVPR Workshop*, 2015. 2

[15] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 1, 4

[16] 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

[17] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012. 1, 4

[18] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 2

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3

[20] P. Tokmakov, K. Alahari, and C. Schmid. Learning motion patterns in videos. *arXiv:1612.07217*, 2016. 2, 4

[21] Y.-H. Tsai, M.-H. Yang, and M. J. Black. Video segmentation via object flow. In *CVPR*, 2016. 2, 4

[22] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 3

[23] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 3