

Lucid Data Dreaming for Multiple Object Tracking

Anna Khoreva · Rodrigo Benenson · Eddy Ilg · Thomas Brox · Bernt Schiele

Received: date / Accepted: date

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 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 [30,32,31] to pixel-level tracking [34,52,46,69]. Given a first frame labelled with the

Anna Khoreva¹ · Rodrigo Benenson² · Eddy Ilg³ · Thomas Brox³ · Bernt Schiele¹

¹Max Planck Institute for Informatics, Germany ²Google

³University of Freiburg, Germany

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.

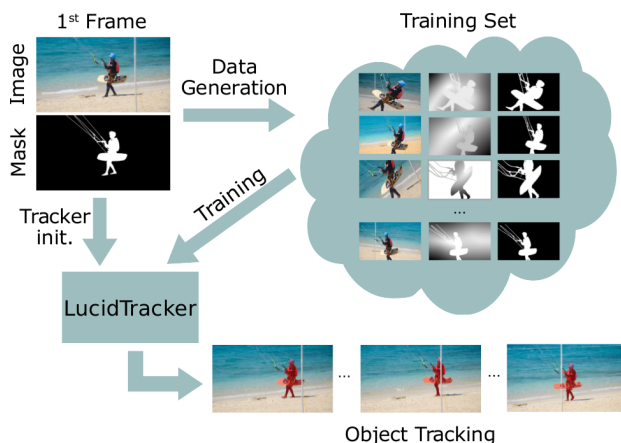


Figure 1: Starting from scarce annotations we synthesize in-domain data to train a specialized pixel-level object tracker for each dataset or even each video.

foreground 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) [27,6,28,3,21,41]. 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 [27,6] or $\sim 10k$ images [28]. 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.

In this work we aim to reduce the necessity for such large volumes of training data. It is traditionally assumed that convnets require 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 [6, 28]. 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 particularly 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 frames of the videos. 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 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 and zero pre-training (i.e. without ImageNet pre-training), we still obtain competitive tracking results.

In summary, our contributions are the following:

1. We propose “*lucid data dreaming*”, an automated approach to synthesize training data for the convnet-based pixel-level object tracking that leads to top results for both single and multiple object tracking.
2. We conduct an extensive analysis to explore the factors contributing to our good results.
3. We show that training a convnet for object tracking can be done with only few annotated frames. We hope these results will affect the trend towards even larger training sets, and popularize the design of trackers with lighter training needs.

2 Related work

Box-level tracking. Classic work on video object tracking focused on bounding box tracking. Many of the insights from these works have been re-used for pixel-level tracking. Traditional box tracking smoothly updates across time a linear model over hand-crafted features [22, 5, 32]. Since then, convnets have been used as improved features [13, 37, 70], and eventually to drive the tracking itself [21, 3, 64, 40, 41]. Contrary to traditional box trackers (e.g. [22]), convnet-based approaches need additional data for pre-training and learning the task.

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

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

Box-to-segment: First a box-level track is built, and a space-time grabcut-like approach is used to generate per frame segments [75].

Video saliency: Instead of tracking, these methods extract the main foreground object pixel-level space-time tube. Both hand-crafted models [16, 43] or trained convnets [65, 26] have been considered. Because these methods ignore the first frame annotation, they fail in videos where multiple salient objects move (e.g. flock of penguins).

Video segmentation methods partition the space-time volume, and then the tube overlapping most with the first frame annotation is selected as tracking output [19, 47, 7].

Mask propagation: Appearance similarity and motion smoothness across time is used to propagate the first frame annotation across the video [38, 72, 66]. These methods usually leverage optical flow and long term trajectories.

Convnets: following the trend in box-level tracking, recently convnets have been proposed for pixel-level tracking. [6] trains a generic object saliency network, and fine-tunes it per-video (using the first frame annotation) to make the output sensitive to the specific object instance being tracked. [28] uses a similar strategy, but also feeds the mask from the previous frame as guidance for the saliency network. Finally [27] mixes convnets with ideas of bilateral filtering. Our approach also builds upon convnets.

What makes convnets particularly suitable for the task, is that they can learn what are the common statistics of appearance and motion patterns of objects, as well as what makes them distinctive from the background, and exploit this knowledge when tracking a particular object. This aspect gives convnets an edge over traditional techniques based on low-level hand-crafted features.

Our network architecture is similar to [6, 28]. Other than implementation details, there are three differentiating factors. One, we use a different strategy for training: [6, 27] rely on consecutive video training frames and [28] 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. Two, our approach exploits optical flow better than these previous methods. Three, we describe an extension to seamlessly handle multiple object tracking.

Interactive video segmentation. Interactive segmentation [39, 25, 59, 71] considers more diverse user inputs (e.g. strokes), and requires interactive processing speed rather than providing maximal quality. Albeit our technique can be adapted for varied inputs, we focus on maximizing quality for the non-interactive case with no-additional hints along the video.

Semantic labelling. Like other convnets in this space [27, 6, 28], our architecture builds upon the insights from the se-

mantic labelling networks [78, 36, 74, 2]. Because of this, the flurry of recent developments should directly translate into better tracking results. For the sake of comparison with previous work, we build upon the well established VGG DeepLab architecture [9].

Synthetic data. Like our approach, previous works have also explored synthesizing training data. Synthetic renderings [42], video game environment [54], mix-synthetic and real images [67, 10, 14] have shown promise, but require task-appropriate 3d models. Compositing real world images provides more realistic results, and has shown promise for object detection [17, 63], text localization [20] and pose estimation [48].

The closest work to ours is [44], 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

Section 3.1 describes the network architecture used, and how RGB and optical flow information are fused to predict the next frame segmentation mask. Section 3.2 discusses different training modalities employed with the proposed object tracking system. In Section 4 we discuss the training data generation, and sections 5/6 report results for single/multiple object tracking.

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 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 flow \mathcal{F}_t enables to 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 . In some experiments we adapt the model f per video, using the annotated first frame \mathcal{I}_0, M_0 . However, in contrast to traditional techniques [22], this model is not updated while we process the video frames,

thus the only state evolving along the video is the mask M_{t-1} itself.

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 sections 5 & 6. If only a bounding box is available on the first frame, then the mask could be estimated using grabcut-like techniques [55, 62].

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 [9], which is comparable to [27, 6, 28]; FusionSeg [26] uses ResNet.

Optical flow \mathcal{F} . We use flow in two complementary ways. First, to obtain a better initial estimate of M_t we warp M_{t-1} using the flow \mathcal{F}_t : $M_t = f_{\mathcal{I}}(\mathcal{I}_t, w(M_{t-1}, \mathcal{F}_t))$; we call this "mask warping". 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 a 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

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

We use the state-of-the-art optical flow estimation method FlowNet2.0 [23], which itself is a convnet that computes $\mathcal{F}_t = h(\mathcal{I}_{t-1}, \mathcal{I}_t)$ and is trained on synthetic renderings of flying objects [42]. For the optical flow magnitude computation we subtract the median motion for each frame, average the magnitude of the forward and backward flow and scale the values per-frame to $[0; 255]$, bringing it to the same range as RGB channels.

The loss function is the sum of cross-entropy terms over each pixel in the output map (all pixels are equally weighted). In our experiments $f_{\mathcal{I}}$ and $f_{\mathcal{F}}$ are trained independently, via some of the modalities listed in Section 3.2. Our two streams 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, see experiments in Section 5.4.3. Our one stream architecture is illustrated in Figure 3b. One stream network is more affordable to train and allows to easily add extra input channels, e.g. providing additionally semantic information about objects.

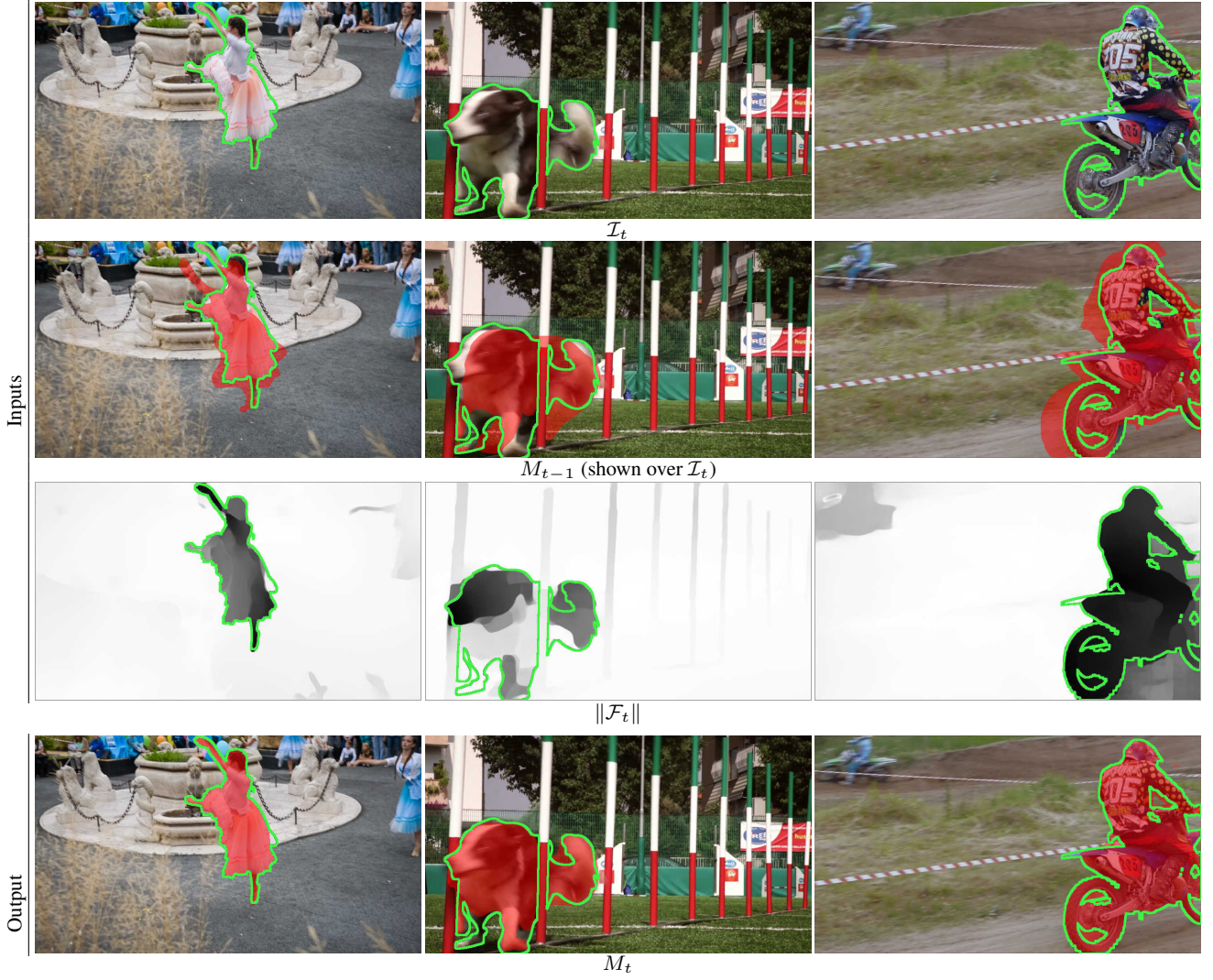


Figure 2: Data flow examples. \mathcal{I}_t , \mathcal{F}_t , M_{t-1} are the inputs, M_t is the resulting output. Green boundaries outline the ground truth segments. Red overlay indicates M_{t-1} , M_t .

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

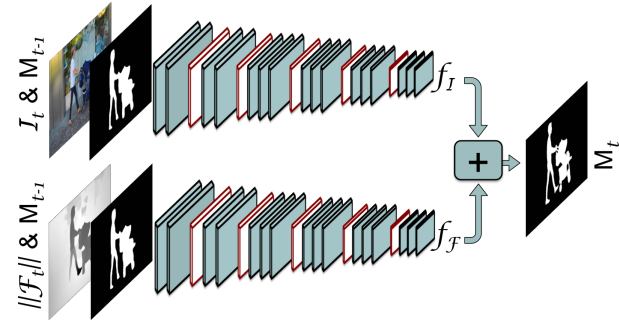
For multiple object tracking task we employ a one-stream architecture for the experiments, using optical flow \mathcal{F} and semantic segmentation \mathcal{S} as additional input channels: $M_t = f_{\mathcal{I}+\mathcal{F}+\mathcal{S}}(\mathcal{I}_t, \mathcal{F}_t, \mathcal{S}_t, w(M_{t-1}^1, \mathcal{F}_t), \dots, w(M_{t-1}^N, \mathcal{F}_t))$. This allows to leverage the appearance model with semantic priors and motion information. See Figure 4 for an illustration.

In our preliminary results using a single architecture provides better results than tracking multiple objects separately, one

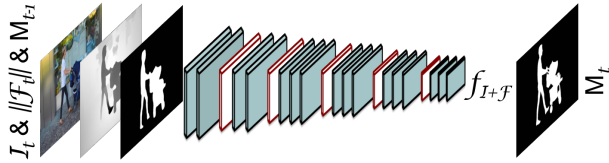
at a time; and avoids the need to design a merging strategy amongst overlapping tracks.

Semantic labels \mathcal{S} . To compute the pixel-level semantic labelling $\mathcal{S}_t = h(\mathcal{I}_t)$ we use the state-of-the-art convnet PSPNet [78], trained on Pascal VOC12 [15]. Pascal VOC12 annotates 20 categories, yet we want to track type of objects. \mathcal{S}_t can also provide information about unknown category instances by describing them as a spatial mixture of known ones (e.g. a sea lion might looks like a dog torso, and the head of cat). As long as the predictions are consistent through time, \mathcal{S}_t will provide a useful cue for tracking. Note that we only use \mathcal{S}_t for the multi-object tracking challenge, discussed in Section 6. In the same way as for the optical flow we scale \mathcal{S}_t to bring all the channels to the same range.

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 mul-



(a) Two streams architecture, where image I_t and optical flow information $\|F_t\|$ are used to update mask M_{t-1} into M_t . See equation 1.



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

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

multiple object tracking task we consider the ensemble of four models: $M_t = 0.25 \cdot (f_{I+F+S} + f_{I+F} + f_{I+S} + f_I)$, 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 per-frame t the generated mask M_t using DenseCRF [29]. This adjusts small image details that the network might not be able to handle. 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 f_{I+F} system as LucidTracker, and as LucidTracker⁻ when no post-processing is used. The usage of S_t or model ensemble will be explicitly stated.

3.2 Training modalities

Multiple modalities are available to train a tracker. **Training-free** approaches (e.g. BVS [38], SVT [72]) 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 [73, 45, 76, 79] or by using the first frame annotation of each video in the dataset as

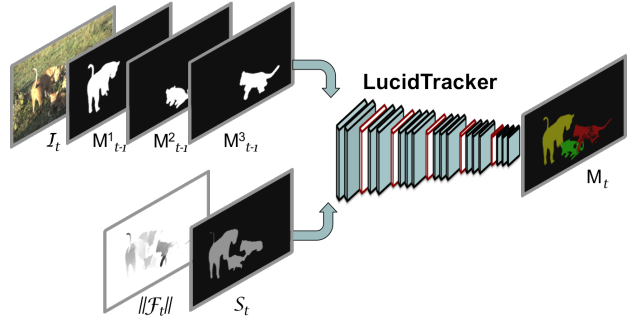


Figure 4: Extension of LucidTracker to multiple objects. The previous frame mask for each object is provided in a separate channel. We additionally explore using optical flow F and semantic segmentation S as additional inputs. See §3.1.

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 new 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. The post-processing DenseCRF stage is automatically tuned per-dataset. The experimental section 5 details the effect of these training stages. Surprisingly, we can obtain reasonable performance even when training from only a single annotated frame (without ImageNet pre-training, i.e. zero pre-training); this results goes against the intuition that convnets require large training data to provide good results.

Unless otherwise stated, we fine-tune per-video models relying solely on the first frame I_0 and its annotation M_0 . This is in contrast to traditional techniques [22, 5, 32] which would update the appearance model at each frame I_t .

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 [3, 6, 21]), however such data is expensive to annotate and rare. [6] thus trains on a set of 30 videos ($\sim 2k$ frames) and requires the model to transfer across multiple tests sets. [28] side-steps the need for consecutive frames by generating synthetic masks M_{t-1} from a 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 individual training frames.

Ideally training data should be as similar as possible to the test data, even subtle differences may affect quality (e.g. training on static images for testing on videos under-

performs [61]). To ensure our training data is in-domain, we propose to generate it by synthesizing samples from the provided annotated frame (first frame) in each target video. This is akin to “lucid dreaming” as we intentionally “dream” the desired data by creating sample images that are plausible hypothetical future frames of the video. The outcome of this process is a large set of frame pairs in the target domain ($2.5k$ pairs per annotation) with known optical flow and mask annotations, see Figure 5.

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 to cover the expected 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 should 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 transformation parameters, resulting in a pair of frames $(\mathcal{I}_{\tau-1}, \mathcal{I}_{\tau})$ with known pixel-level ground-truth mask annotations $(M_{\tau-1}, M_{\tau})$, optical flow \mathcal{F}_{τ} , and occlusion regions. The object position in \mathcal{I}_{τ} 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 [12].
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}_{τ} with a translation of $\pm 10\%$ of the object size relative to $\tau - 1$. In both frames we apply random rotation $\pm 30^\circ$, scaling $\pm 15\%$ and thin-plate splines deformations [4] 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 [60]. Using the known transformation parameters we also synthesize ground-truth pixel-level mask annotations $(M_{\tau-1}, M_{\tau})$ and optical flow \mathcal{F}_{τ} .

Figure 5 shows example results. Albeit our approach does

not capture appearance changes due to point of view, occlusions, nor shadows, we see that already this rough modelling is effective to train our tracking models.

The number of synthesized images can be arbitrarily large. We generate $2.5k$ pairs per annotated video frame. This training data is, by design, in-domain with regard of the target video. The experimental section 5 shows that this strategy is more effective than using thousands of manually annotated images from close-by domains.

The same strategy for data synthesis can be employed for multiple object tracking task. Instead of manipulating a single object we handle multiple ones at the same time, applying independent transformations to each of them. We model occlusion between objects by adding a random depth ordering obtaining both partial and full occlusions in the training set. Including occlusions in the lucid dreams allows to better handle plausible interactions of objects in the future frames. See Figure 6 for examples of the generated data.

5 Single object tracking results

We present here a detailed empirical evaluation on three different datasets for the single object tracking task: given a first frame labelled with the foreground object mask, the goal is to find the corresponding object pixels in future frames. (Section 6 will discuss the multiple objects case.)

5.1 Experimental setup

Datasets. We evaluate our method on three video object segmentation datasets: DAVIS₁₆ [46], YouTubeObjects [52, 24], and SegTrack_{v2} [34]. The goal is to track an object through all video frames given a foreground object mask in the first frame. These three datasets provide diverse challenges with a mix of high and low resolution web videos, single or multiple salient objects per video, videos with flocks of similar looking instances, longer (~ 400 frames) and shorter (~ 10 frames) sequences, as well as the usual tracking challenges such as occlusion, fast motion, illumination, view point changes, elastic deformation, etc.

The DAVIS₁₆ [46] video segmentation benchmark consists of 50 full-HD videos of diverse object categories with all frames annotated with pixel-level accuracy, where one single or two connected moving objects are separated from the background. The number of frames in each video varies from 25 to 104.

YouTubeObjects [52, 24] includes web videos from 10 object categories. We use the subset of 126 video sequences with mask annotations provided by [24] for evaluation, where one single object or a group of objects of the same category are separated from the background. In contrast to DAVIS₁₆

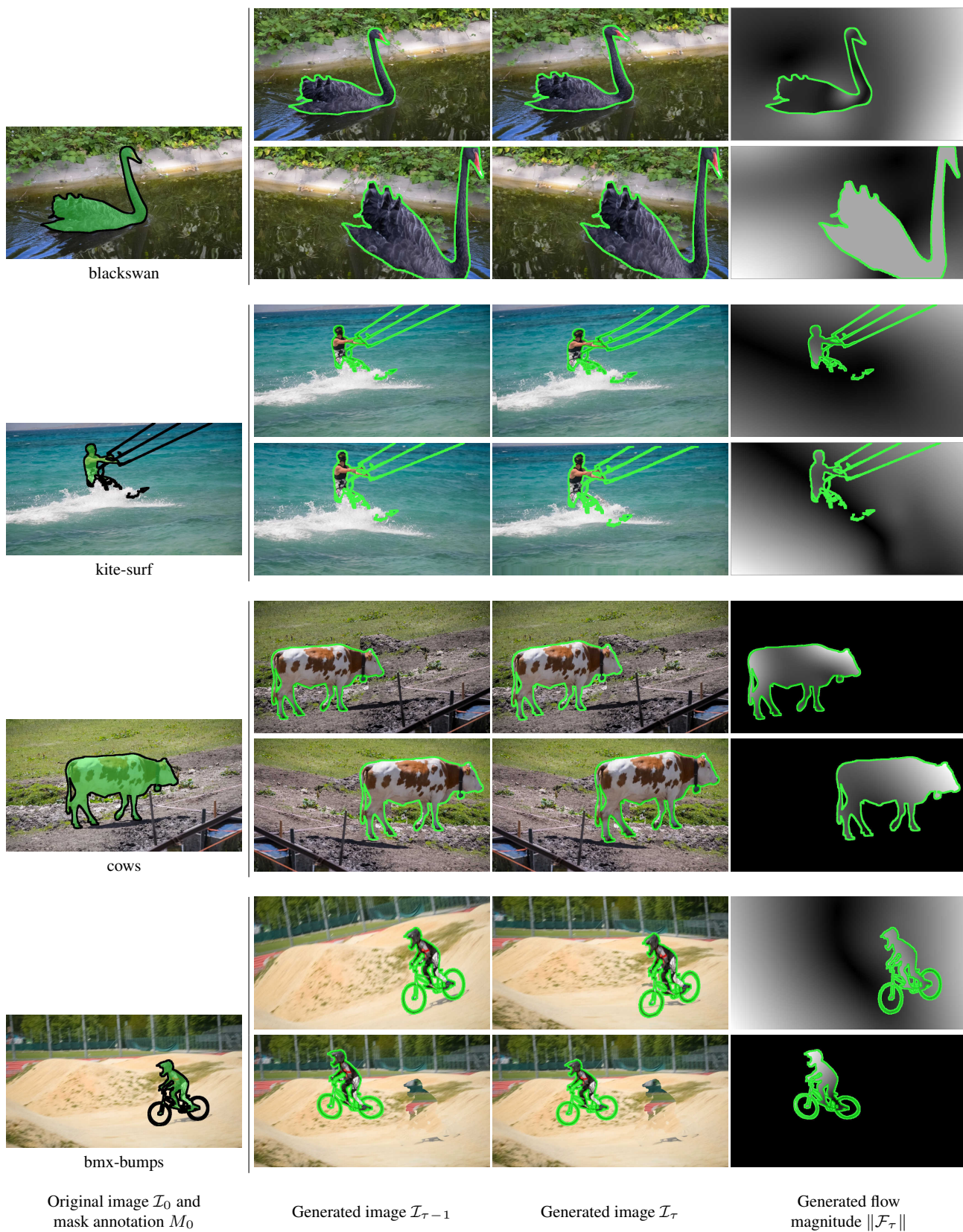


Figure 5: Lucid data dreaming examples. From one annotated frame we generate pairs of images ($\mathcal{I}_{\tau-1}$, \mathcal{I}_{τ}) that are plausible future video frames, with known optical flow (\mathcal{F}_{τ}) and masks (green boundaries). Note the inpainted background and foreground/background deformations.

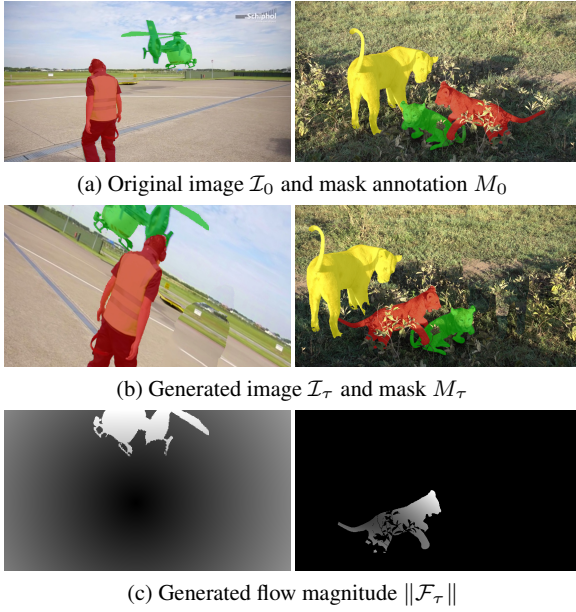


Figure 6: Lucid data dreaming examples with multiple objects. From one annotated frame we generate a plausible future video frame (\mathcal{I}_τ), with known optical flow (\mathcal{F}_τ) and mask (M_τ).

these videos have a mix of static and moving objects. The number of frames in each video ranges from 2 to 401.

SegTrack_{v2}[34] consists of 14 videos with multiple object annotations for each frame. For videos with multiple objects each object is treated as a separate problem, resulting in 24 sequences. The length of each video varies from 21 to 279 frames. The images in this dataset have low resolution and some compression artefacts, making it hard to track the object based on its appearance.

The main experimental work is done on DAVIS₁₆, since it is the largest densely annotated dataset out of the three, and provides high quality/high resolution data. The videos for this dataset were chosen to represent diverse challenges, making it a good experimental playground.

We additionally report on the two other datasets as complementary test set results.

Evaluation metric. To measure the accuracy of video object tracking we use the mean intersection-over-union overlap (mIoU) between the per-frame ground truth object mask and the predicted segmentation, averaged across all video sequences. We have noticed disparate evaluation procedures used in previous work, and we report here a unified evaluation across datasets. When possible, we re-evaluated certain methods using results provided by their authors. For all three datasets we follow the DAVIS₁₆ evaluation protocol, excluding the first frame from evaluation and using all other frames from the video sequences, independent of object presence in the frame.

Training details. For training all the models we use SGD with mini-batches of 10 images and a fixed learning policy with initial learning rate of 10^{-3} . The momentum and weight decay are set to 0.9 and $5 \cdot 10^{-4}$, respectively.

Models using pre-training are initialized with weights trained for image classification on ImageNet [58]. We then train per-dataset for 40k iterations with the RGB+Mask branch $f_{\mathcal{I}}$ and for 20k iterations for the Flow+Mask $f_{\mathcal{F}}$ branch. When using a single stream architecture (Section 5.4.3), we use 40k iterations.

Models without ImageNet pre-training are initialized using the “Xavier” strategy [18]. The per-dataset training needs to be longer, using 100k iterations for the $f_{\mathcal{I}}$ branch and 40k iterations for the $f_{\mathcal{F}}$ branch.

For per-video fine-tuning 2k iterations are used for $f_{\mathcal{I}}$. To keep computing cost lower, the $f_{\mathcal{F}}$ branch is kept fix across videos.

All training parameters are chosen based on DAVIS₁₆ results. We use identical parameters on YouTubeObjects and SegTrack_{v2}, showing the generalization of our approach.

It takes ~ 3.5 h to obtain each per-video model, including data generation, per-dataset training, per-video fine-tuning and per-dataset grid search of CRF parameters (averaged over DAVIS₁₆, amortising the per-dataset training time over all videos). At test time our LucidTracker runs at ~ 5 s per frame, including the optical flow estimation with FlowNet2.0 [23] (~ 0.5 s) and CRF post-processing [29] (~ 2 s).

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 DAVIS₁₆, 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.

Oracles and baselines. Grabcut oracle computes grabcut [55] using the ground truth bounding boxes (box oracle). This oracle indicates that on the considered datasets separating foreground from background is not easy, even if a perfect box-level tracker was available.

We provide three additional baselines. “Saliency” corresponds to using the generic (training-free) saliency method EQCut [1] over the RGB image \mathcal{I}_t . “Flow saliency” does the same, but over the optical flow magnitude $\|\mathcal{F}_t\|$. Results indicate that the objects being tracked are not particularly salient in the image. On DAVIS₁₆ motion saliency is a strong signal but not on the other two datasets. Saliency methods ignore the first frame annotation provided for the tracking task. We also consider the “Mask warping” baseline which

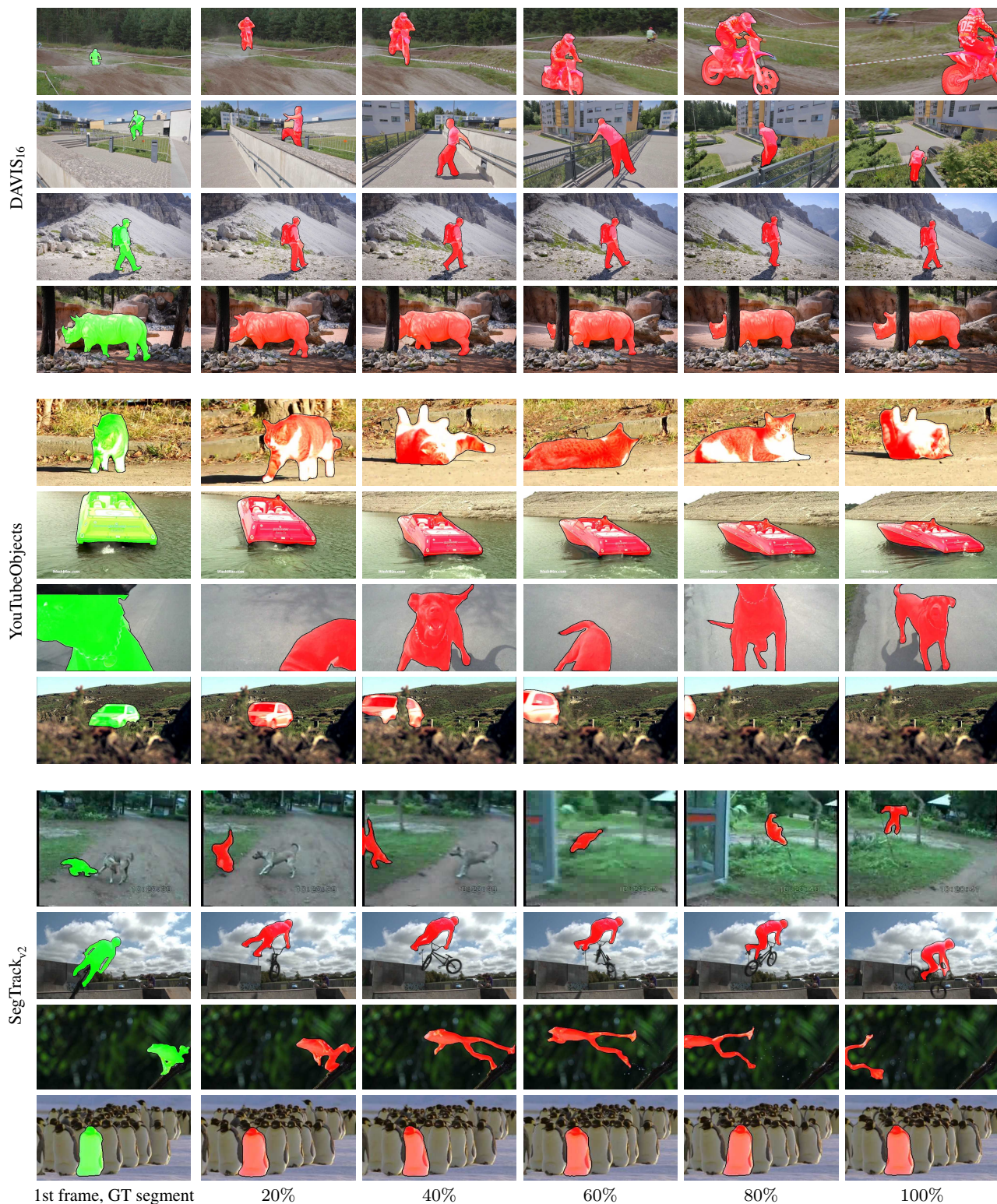


Figure 7: LucidTracker single object tracking qualitative results. Frames sampled along the video duration (e.g. 50%: video middle point). Our model is robust to various challenges, such as view changes, fast motion, shape deformations, and out-of-view scenarios.

	Method	# training Flow images	\mathcal{F}	Dataset, mIoU		
				DAVIS ₁₆	YoutbObjs	SegTrck _{v2}
Ignores 1st frame annotation	Box oracle [28]	0	✗	45.1	55.3	56.1
	Grabcut oracle [28]	0	✗	67.3	67.6	74.2
	Saliency	0	✗	32.7	40.7	22.2
	NLC [16]	0	✓	64.1	-	-
	TRS [75]	0	✓	-	-	69.1
	MP-Net [65]	~22.5k	✓	69.7	-	-
	Flow saliency	0	✓	70.7	36.3	35.9
Uses 1st frame annotation	FusionSeg [26]	~95k	✓	71.5	67.9	-
	Mask warping	0	✓	32.1	43.2	42.0
	FCP [47]	0	✓	63.1	-	-
	BVS [38]	0	✗	66.5	59.7	58.4
	N15 [39]	0	✓	-	-	69.6
	ObjFlow [66]	0	✓	71.1	70.1	67.5
	STV [72]	0	✓	73.6	-	-
	VPN [27]	~2.3k	✗	75.0	-	-
	OSVOS [6]	~2.3k	✗	79.8	72.5	65.4
	MaskTrack [28]	~11k	✓	80.3	72.6	70.3
	LucidTracker	24~126	✓	84.8	76.2	77.6

Table 1: Comparison of segment tracking results across three datasets. Numbers in *italic* are reported on subsets of DAVIS₁₆. Our LucidTracker consistently improves over previous results, see §5.2.

uses optical flow to propagate the mask estimate from t to $t + 1$ via simple warping $M_t = w(M_{t-1}, \mathcal{F}_t)$. The bad results of this baseline indicate that the high quality flow [23] that we use is by itself insufficient to solve the tracking task, and that indeed our proposed convnet does the heavy lifting.

The large fluctuation of the relative baseline results across the three datasets empirically confirms that each of them presents unique challenges.

Comparison. Compared to flow propagation methods such as BVS, N15, ObjFlow, and STV, we obtain better results because 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.

Figure 7 provides qualitative results of LucidTracker across three different datasets. Our system is robust to various challenges present in videos. It handles well camera view changes, fast motion, object shape deformation, out-of-view scenarios, multiple similar looking objects and even low quality video. We provide a detailed error analysis in section 5.5.

Conclusion. We show that top results can be obtained while using less training data. This shows that our lucid dreams leverage the available training data better. We report top results for this task while using only 24~126 training frames.

Variant	ImgNet	per-dataset	per-video	Dataset, mIoU		
	pre-train.	training	fine-tun.	DAVIS ₁₆	YoutbObjs	SegTrck _{v2}
LucidTracker ⁻	✓	✓	✓	83.7	76.2	76.8
(no ImgNet)	✗	✓	✓	82.0	74.3	71.2
No per-video tuning	✓	✓	✗	82.7	72.3	71.9
	✗	✓	✗	78.4	69.7	68.2
Only per-video tuning	✓	✗	✓	79.4	-	70.4
	✗	✗	✓	80.5	-	66.8

Table 2: Ablation study of training modalities. ImageNet pre-training and per-video tuning provide additional improvement over per-dataset training. Even with one frame annotation for only per-video tuning we obtain good performance. See §5.3.1.

5.3 Ablation studies

In this section we explore in more details how the different ingredients contribute to our results.

5.3.1 Effect of training modalities

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

Training from a single frame. 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*. Our network is based on VGG16 [9] and contains $\sim 20M$ parameters, all effectively learnt from a single annotated image that is augmented to become 2.5k training samples (see Section 4). Even with such minimal amount of training data, we still obtain a surprisingly good performance (compare 80.5 on DAVIS₁₆ to others in Table 1). This shows how effective is, by itself, the proposed training strategy based on lucid dreaming of the data.

Pre-training & fine-tuning. We see that ImageNet pre-training does provide 2~5 percent point improvement (depending on the dataset of interest; e.g. 82.0 \rightarrow 83.7 mIoU on DAVIS₁₆). Per-video fine-tuning (after doing per-dataset training) provides an additional 1~2 percent point gain (e.g. 82.7 \rightarrow 83.7 mIoU on DAVIS₁₆). Both ingredients clearly contribute to the tracking results.

Note that training a model using only per-video tuning takes about one full GPU day per video sequence; making these results insightful but not decidedly practical.

Preliminary experiments evaluating on DAVIS₁₆ the impact of the different ingredients of our lucid dreaming data generation showed, depending on the exact setup, 3~10 percent mIoU points fluctuations between a basic version

Variant	\mathcal{I} \mathcal{F} warp. per-video fine-tun.				Dataset, mIoU		
	w	\mathcal{F}	\mathcal{F}	\mathcal{F}	DAVIS ₁₆	YoutbObjs	SegTrck _{v2}
LucidTracker	✓	✓	✓	✗	84.8	76.2	77.6
LucidTracker ⁻	✓	✓	✓	✗	83.7	76.2	76.8
No warping	✓	✓	✗	✗	82.0	74.6	70.5
No OF	✓	✗	✗	✗	78.0	74.7	61.8
OF only	✗	✓	✓	✗	74.5	43.1	55.8

Table 3: Ablation study of flow ingredients. Flow complements image only results, with large fluctuations across datasets. See §5.3.2.

(e.g. without non-rigid deformations nor scene re-composition) and the full synthesis process described in Section 4. Having a sophisticated data generation process directly impacts the tracking quality.

Conclusion. Surprisingly, we discovered that per-video training from a single annotated frame provides already much of the information needed for the tracking task. Additionally using ImageNet pre-training, and per-dataset training, provide complementary gains.

5.3.2 Effect of optical flow

Table 3 shows the effect of optical flow on LucidTracker results. Comparing our full system to the "No OF" row, we see that the effect of optical flow varies across datasets, from minor improvement in YouTubeObjects, to major difference in SegTrack_{v2}. In this last dataset, using mask warping is particularly useful too. We additionally explored tuning the optical flow stream per-video, which resulted in a minor improvement (83.7 → 83.9 mIoU on DAVIS₁₆).

Our "No OF" results can be compared to OSVOS [6] which does not use optical flow. However OSVOS uses a per-frame mask post-processing based on a boundary detector (trained on further external data), which provides ~2 percent point gain. Accounting for this, our "No OF" (and no CRF) result matches theirs on DAVIS₁₆ and YouTubeObjects despite using significantly less training data (see Table 1, e.g. 79.8 − 2 ≈ 78.0 on DAVIS₁₆).

Table 4 shows the effect of using different optical flow estimation methods. For LucidTracker results, FlowNet2.0 [23] was employed. We also explored using EpicFlow [53], as in [28]. Table 4 indicates that employing a robust optical flow estimation across datasets is crucial to the performance (FlowNet2.0 provides ~ 1.5 – 15 points gain on each dataset). We found EpicFlow to be brittle when going across different datasets, providing improvement for DAVIS₁₆ and SegTrack_{v2} (~ 2 – 5 points gain), but underperforming for YouTubeObjects (74.7 → 71.3 mIoU).

Variant	Optical flow	Dataset, mIoU		
		DAVIS ₁₆	YoutbObjs	SegTrck _{v2}
LucidTracker ⁻	FlowNet2.0	83.7	76.2	76.8
	EpicFlow	80.2	71.3	67.0
	No flow	78.0	74.7	61.8
No ImageNet pre-training	FlowNet2.0	82.0	74.3	71.2
	EpicFlow	80.0	72.3	68.8
	No flow	76.7	71.4	63.0

Table 4: Effect of optical flow estimation.

Method	CRF parameters	Dataset, mIoU		
		DAVIS ₁₆	YoutbObjs	SegTrck _{v2}
LucidTracker ⁻	-	83.7	76.2	76.8
LucidTracker	default	84.2	75.5	72.2
LucidTracker	tuned per-dataset	84.8	76.2	77.6

Table 5: Effect of CRF tuning. Without per-dataset tuning DenseCRF will under-perform.

Conclusion. The results show that flow provides a complementary signal to RGB image only and having a robust optical flow estimation across datasets is crucial. Despite its simplicity our fusion strategy ($f_{\mathcal{I}} + f_{\mathcal{F}}$) provides gains on all datasets, and leads to competitive results.

5.3.3 Effect of CRF tuning

As a final stage of our pipeline, we refine the generated mask using DenseCRF [29] per frame. This captures small image details that the network might have missed. It is known by practitioners that DenseCRF is quite sensitive to its parameters and can easily worsen results. We use our lucid dreams to enable automatic per-dataset CRF-tuning.

Following [9] we employ grid search scheme for tuning CRF parameters. Once the per-dataset tracking model is trained, we apply it over a subset of its training set (5 random images from the lucid dreams per video sequence), apply DenseCRF with the given parameters over this output, and then compare to the lucid dream ground truth.

The impact of the tuned parameter of DenseCRF post-processing is shown in Table 5 and Figure 8. Table 5 indicates that without per-dataset tuning DenseCRF is under-performing. Our automated tuning procedure allows to obtain consistent gains without the need for case-by-case manual tuning.

Conclusion. Using default DenseCRF parameters will degrade performance. Our lucid dreams enable per-dataset CRF-tuning which allows to further improve the results.

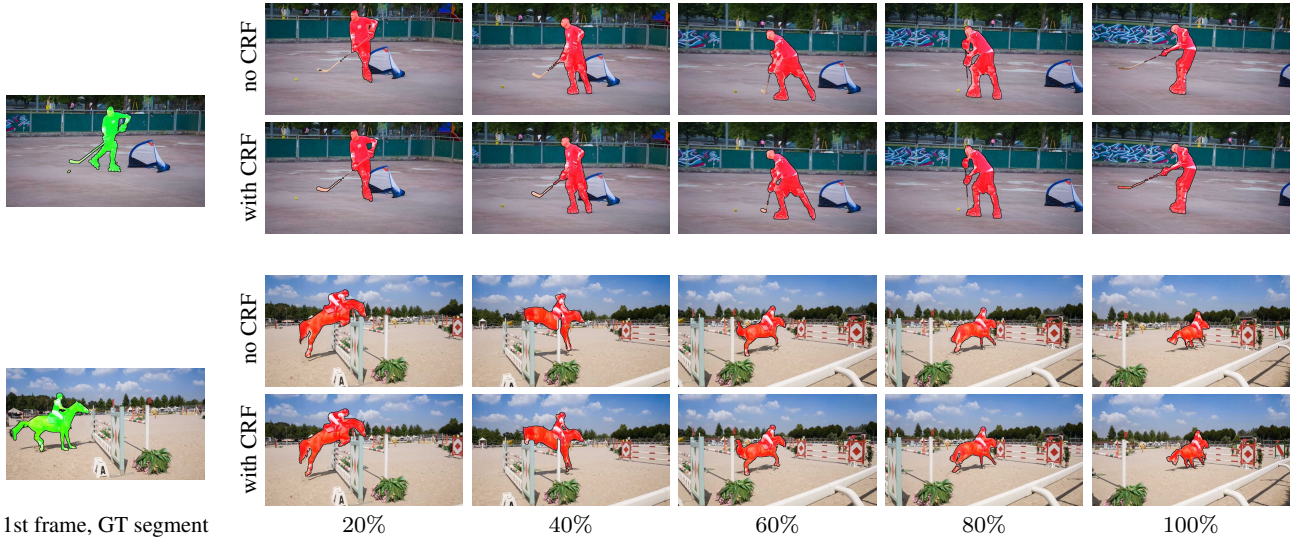


Figure 8: Effect of CRF tuning. The shown DAVIS₁₆ videos have the highest margin between with and without CRF post-processing (based on mIoU over the video).

Training set	# training videos	# frames per video	mIoU
Includes 1st frames from test set	1	1	78.3
	2	1	75.4
	15	1	68.7
	30	1	65.4
	30	2	74.3
Excludes 1st frames from test set	2	1	11.6
	15	1	36.4
	30	1	41.7
	30	2	48.4

Table 6: Varying the number of training videos. A smaller training set closer to the target domain is better than a larger one. See Section 5.4.1.

5.4 Additional experiments

Other than adding or removing ingredients, as in Section 5.3, we also want to understand how the training data itself affects the obtained results.

5.4.1 Generalization across videos

Table 6 explores the effect of tracking quality as a function of the number of training samples. To see more directly the training data effects we use a base model with RGB image \mathcal{I}_t only (no flow \mathcal{F} , no CRF), and per-dataset training (no ImageNet pre-training, no per-video fine-tuning). We evaluate on two disjoint subsets of 15 DAVIS₁₆ videos each, where the first frames for per-dataset training are taken from only one subset. The reported numbers are thus comparable

within Table 6, but not across to the other tables in the paper. Table 6 reports results with varying number of training videos and with/without including the first frames of each test video for per-dataset training. When excluding the test set first frames, the image frames used for training are separate from the test videos; and we are thus operating across (related) domains. When including the test set first frames, we operate in the usual LucidTracker mode, where the first frame from each test video is used to build the per-dataset training set.

Comparing the top and bottom parts of the table, we see that when the annotated images from the test set videos are not included, tracking quality drops drastically (e.g. 68.7 \rightarrow 36.4 mIoU). Conversely, on subset of videos for which the first frame annotation is used for training, the quality is much higher and improves as the training samples become more and more specific (in-domain) to the target video (65.4 \rightarrow 78.3 mIoU). Adding extra videos for training does not improve the performance. It is better (68.7 \rightarrow 78.3 mIoU) to have 15 models each trained and evaluated on a single video (row top-1-1) than having one model trained over 15 test videos (row top-15-1).

Training with an additional frame from each video (we added the last frame of each train video) significantly boosts the resulting within-video quality (e.g. row top-30-2 65.4 \rightarrow 74.3 mIoU), because the training samples cover better the test domain.

Conclusion. These results show that, when using RGB information (\mathcal{I}_t), increasing the number of training videos *does not* improve the resulting quality of our system. Even within a dataset, properly using the training sample(s) from within

Training set	Dataset, mIoU			Mean
	DAVIS ₁₆	YoutbObjs	SegTrack _{v2}	
DAVIS ₁₆	<u>80.9</u>	50.9	46.9	59.6
YoutbObjs	67.0	<u>71.5</u>	52.0	63.5
SegTrack _{v2}	56.0	52.2	<u>66.4</u>	58.2
Best	80.9	71.5	66.4	72.9
Second best	67.0	52.2	52.0	57.1
All-in-one	71.9	70.7	60.8	67.8

Table 7: Generalization across datasets. Results with underline are the best per dataset, and in italic are the second best per dataset (ignoring all-in-one setup). We observe a significant quality gap between training from the target videos, versus training from other datasets; see §5.4.2.

each video matters more than collecting more videos to build a larger training set.

5.4.2 Generalization across datasets

Section 5.4.1 has explored the effect of changing the volume of training data within one dataset, Table 7 compares results when using different datasets for training. Results are obtained using a base model with RGB and flow ($M_t = f(\mathcal{I}_t, M_{t-1})$, no warping, no CRF), ImageNet pre-training, per-dataset training, and no per-video tuning to accentuate the effect of the training dataset.

The best performance is obtained when training on the first frames of the target set. There is a noticeable ~ 10 percent points drop when moving to the second best choice (e.g. $80.9 \rightarrow 67.0$ for DAVIS₁₆). Interestingly, when putting all the datasets together for training ("all-in-one" row, a dataset-agnostic model) the results degrade, reinforcing the idea that "just adding more data" does not automatically make the performance better.

Conclusion. Best results are obtained when using training data that focuses on the test video sequences, using similar datasets or combining multiple datasets degrades the performance for our system.

5.4.3 Experimenting with the convnet architecture

Section 3.1 and Figure 3 described two possible architectures to handle \mathcal{I}_t and \mathcal{F}_t . Previous experiments are all based on the two streams architecture.

Table 8 compares two streams versus one stream, where the network to accepts 5 input channels (RGB + previous mask + flow magnitude) in one stream: $M_t = f_{\mathcal{I}+\mathcal{F}}(\mathcal{I}_t, \mathcal{F}_t, w(M_{t-1}, \mathcal{F}_t))$. Results are obtained using a base model with RGB and optical flow (no warping, no CRF), ImageNet pre-training, per-dataset training, and no per-video tuning.

We observe that both one stream and two stream architecture with naive averaging perform on par. Using a one

Architecture	ImgNet pre-train.	per-dataset training	per-video fine-tun.	DAVIS ₁₆ mIoU
two streams	✓	✓	✗	80.9
one stream	✓	✓	✗	80.3

Table 8: Experimenting with the convnet architecture. See §5.4.3.

stream network makes the training more affordable and allows more easily to expand the architecture with additional input channels.

Conclusion. The lighter one stream network performs as well as a network with two streams. We will thus use the one stream architecture in Section 6.

5.5 Error analysis

Table 9 presents an expanded evaluation on DAVIS₁₆ using evaluation metrics proposed in [46]. Three measures are used: region similarity in terms of intersection over union (J), contour accuracy (F, higher is better), and temporal instability of the masks (T, lower is better). We outperform the competitive methods [28, 6] on all three measures.

Table 10 reports the per-attribute based evaluation as defined in DAVIS₁₆. LucidTracker is best on 13 out of 15 video attribute categories. This shows that LucidTracker can handle various video challenges present in DAVIS₁₆.

We present the per-sequence and per-frame results of LucidTracker over DAVIS₁₆ in Figure 10. On the whole we observe that the proposed approach is quite robust, most video sequences reach an average performance above 80 mIoU.

However, by looking at per-frame results for each video (blue dots in Figure 10) one can see several frames where our approach has failed (IoU less than 50) to correctly track the object. Investigating closely those cases we notice conditions where LucidTracker is more likely to fail. The same behaviour was observed across all three datasets. A few representatives of failure cases are visualized in Figure 9.

Since we are using only the annotation of the first frame for training the tracker, a clear failure case is caused by dramatic view point changes of the object from its first frame appearance, as in row 5 of Figure 9. The proposed approach also under-performs when recovering from occlusions: it takes several frames for the full object mask to re-appear (rows 1-2 in Figure 9). This is mainly due to the convnet having learnt to follow-up the previous frame mask. Augmenting the lucid dreams with plausible occlusions might help mitigate this case. Another failure case occurs when two similar looking objects cross each other, as in row 6 in Figure 9. Here both cues: the previous frame guidance and learnt via per-video

	Method	# training images	Flow \mathcal{F}	DAVIS ₁₆						
				Region, J			Boundary, F'			Temporal stability, T
				Mean \uparrow	Recall \uparrow	Decay \downarrow	Mean \uparrow	Recall \uparrow	Decay \downarrow	Mean \downarrow
	Box oracle [28]	0	✗	45.1	39.7	-0.7	21.4	6.7	1.8	1.0
	Grabcut oracle [28]	0	✗	67.3	76.9	1.5	65.8	77.2	2.9	34.0
Ignores 1st frame annotation	Saliency	0	✗	32.7	22.6	-0.2	26.9	10.3	0.9	32.8
	NLC [16]	0	✓	64.1	73.1	8.6	59.3	65.8	8.6	35.8
	MP-Net [65]	~22.5k	✓	69.7	82.9	5.6	66.3	78.3	6.7	68.6
	Flow saliency	0	✓	70.7	83.2	6.7	69.7	82.9	7.9	48.2
	FusionSeg [26]	~95k	✓	71.5	-	-	-	-	-	-
Uses 1st frame annotation	Mask warping	0	✓	32.1	25.5	31.7	36.3	23.0	32.8	8.4
	FCP [47]	0	✓	63.1	77.8	3.1	54.6	60.4	3.9	28.5
	BVS [38]	0	✗	66.5	76.4	26.0	65.6	77.4	23.6	31.6
	ObjFlow [66]	0	✓	71.1	80.0	22.7	67.9	78.0	24.0	22.1
	STV [72]	0	✓	73.6	-	-	72.0	-	-	-
	VPN [27]	~2.3k	✗	75.0	-	-	72.4	-	-	29.5
	OSVOS [6]	~2.3k	✗	79.8	93.6	14.9	80.6	92.6	15.0	37.6
	MaskTrack [28]	~11k	✓	80.3	93.5	8.9	75.8	88.2	9.5	18.3
	LucidTracker	24~126	✓	84.8	94.6	4.3	82.3	90.5	7.0	15.8

Table 9: Comparison of segment tracking results on DAVIS₁₆ benchmark. Numbers in *italic* are computed based on subsets of DAVIS₁₆. Our LucidTracker improves over previous results.

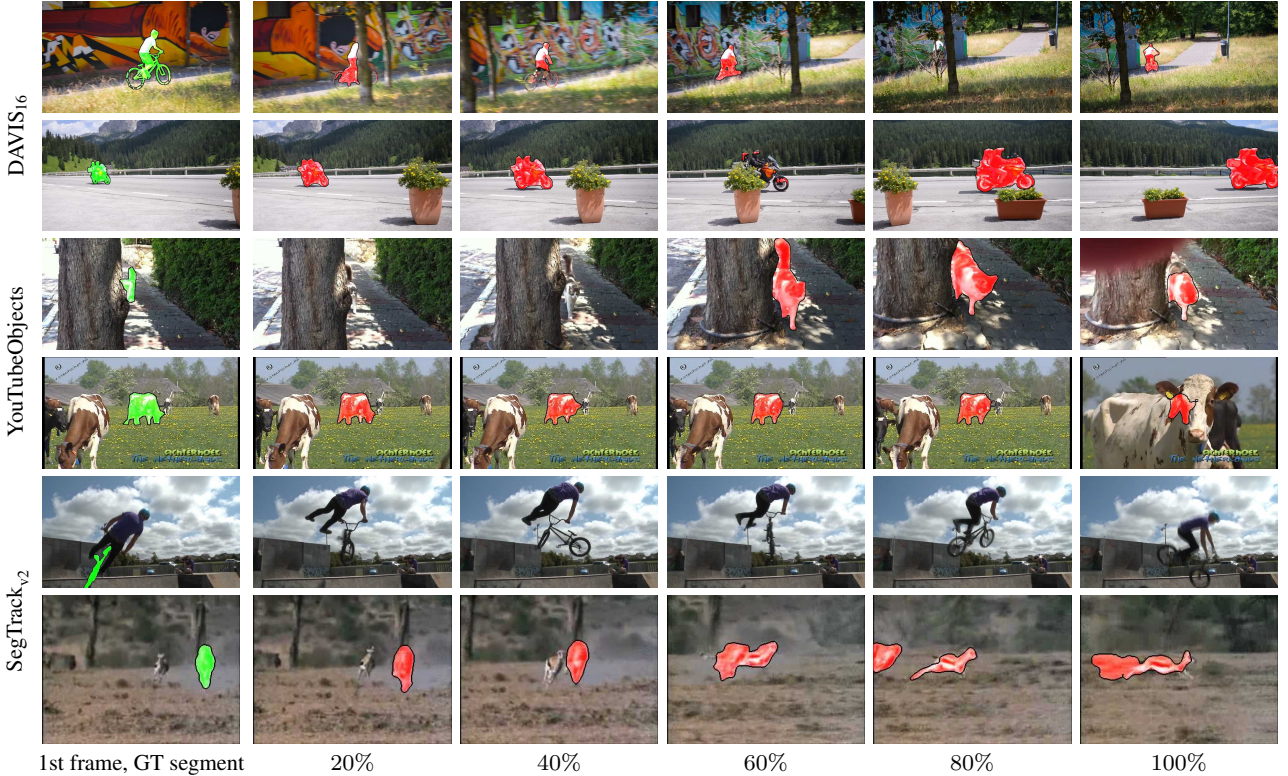
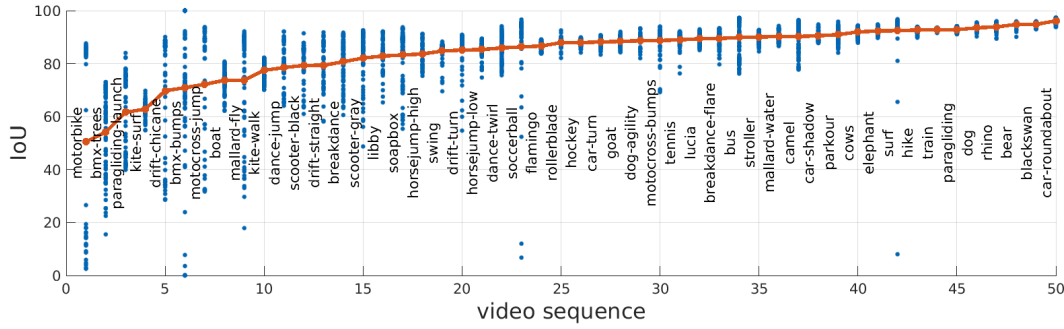


Figure 9: Failure cases. Frames sampled along the video duration (e.g. 50%: video middle point). For each dataset we show 2 out of 5 worst results (based on mIoU over the video).

Figure 10: Per-sequence results on DAVIS₁₆.

Attribute	Method				
	BVS [38]	ObjFlow [66]	OSVOS [6]	MaskTrack [28]	LucidTracker
Appearance change	0.46	0.54	0.81	0.76	0.78
Background clutter	0.63	0.68	0.83	0.79	0.85
Camera-shake	0.62	0.72	0.78	0.78	0.87
Deformation	0.7	0.77	0.79	0.78	0.87
Dynamic background	0.6	0.67	0.74	0.76	0.77
Edge ambiguity	0.58	0.65	0.77	0.74	0.78
Fast-motion	0.53	0.55	0.76	0.75	0.80
Heterogeneous object	0.63	0.66	0.75	0.79	0.83
Interacting objects	0.63	0.68	0.75	0.77	0.84
Low resolution	0.59	0.58	0.77	0.77	0.76
Motion blur	0.58	0.6	0.74	0.74	0.83
Occlusion	0.68	0.66	0.77	0.77	0.83
Out-of-view	0.43	0.53	0.72	0.71	0.84
Scale variation	0.49	0.56	0.74	0.73	0.76
Shape complexity	0.67	0.69	0.71	0.75	0.81

Table 10: DAVIS₁₆ per-attribute evaluation. LucidTracker improves across the bulk of tracking challenges.

tuning appearance, are no longer discriminative to correctly continue tracking.

We also observe that the LucidTracker struggles to track the fine structures or details of the object, e.g. wheels of the bicycle or motorcycle in rows 1-2 in Figure 9. This is the issue of the underlying choice of the convnet architecture, due to the several pooling layers the spatial resolution is lost and hence the fine details of the object are missing. This issue can be mitigated by switching to more recent semantic labelling architectures (e.g. [49, 8]).

Conclusion. LucidTracker shows robust performance across different videos. However, a few failure cases were observed due to the underlying convnet architecture, its training, or limited visibility of the object in the first frame.

6 Multiple object tracking results

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 to find the corresponding masks of objects in future frames.

6.1 Experimental setup

Dataset. For multiple object tracking we use the 2017 DAVIS Challenge on Video Object Segmentation² [51] (DAVIS₁₇). Compared to DAVIS₁₆ this is a larger, more challenging dataset, where the video sequences have multiple objects in the scene. Videos that have more than one visible object in DAVIS₁₆ have been re-annotated (the objects were divided by semantics) and the train and val sets were extended with more sequences. In addition, two other test sets (test-dev and test-challenge) were introduced. The complexity of the videos has increased with more distractors, occlusions, fast motion, smaller objects, and fine structures. Overall, DAVIS₁₇ consists of 150 sequences, totalling 10 474 annotated frames and 384 objects.

We evaluate our method on two test sets, the test-dev and test-challenge sets, each consists of 30 video sequences, on average ~ 3 objects per sequence, the length of the sequences is ~ 70 frames. For both test sets only the masks on the first frames are made public, the evaluation is done via an evaluation server.

Our experiments and ablation studies are done on the test-dev set.

Evaluation metric. The accuracy of multiple object tracking is evaluated using the region (J) and boundary (F) measures proposed by the organisers of the challenge. The average of J and F measures is used as overall performance score. Please refer to [51] for more details about the evaluation protocol.

Training details. All experiments in this section are done using the single stream architecture discussed in sections 3.1 and 5.4.3. For training the models we use SGD with mini-batches of 10 images and a fixed learning policy with initial learning rate of 10^{-3} . The momentum and weight decay are set to 0.9 and $5 \cdot 10^{-4}$, respectively. All models are initialized with weights trained for image classification on ImageNet [58]. We then train per-video for 40k iterations.

² <http://davischallenge.org/challenge2017>

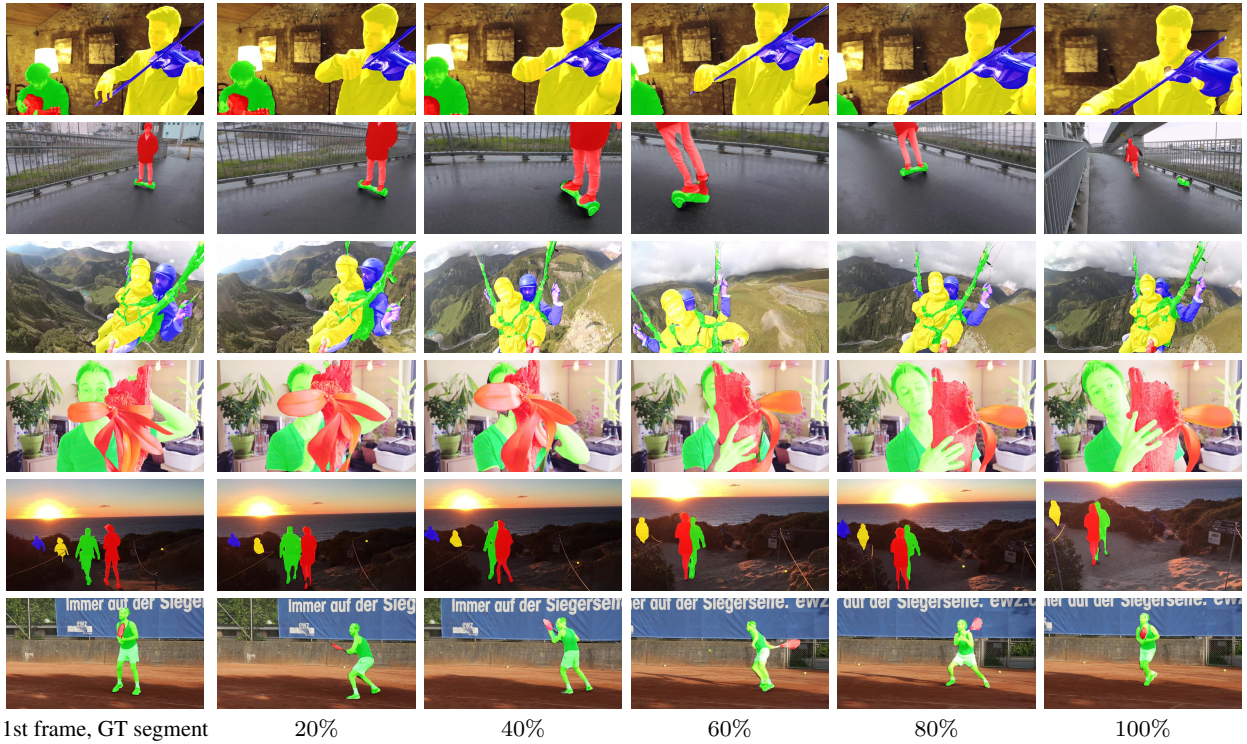


Figure 11: LucidTracker qualitative results on DAVIS₁₇, test-dev set. Frames sampled along the video duration (e.g. 50%: video middle point). The videos are chosen with the highest mIoU measure.

6.2 Key results

Tables 11 and 12 presents the results of the 2017 DAVIS Challenge on test-dev and test-challenge sets [50].

Our main results for the multi-object tracking challenge are obtained via an ensemble of four different models (f_I , f_{I+F} , f_{I+S} , f_{I+F+S}), see Section 3.1.

The proposed system, LucidTracker, provides the best tracking quality on the test-dev set and shows competitive performance on the test-challenge set, holding the second place in the competition. The full system is trained using the standard ImageNet pre-training initialization, Pascal VOC12 semantic annotations for the S_t input ($\sim 10k$ annotated images), and one annotated frame per test video, 30 frames total on each test set. As discussed in Section 6.3, even without S_t LucidTracker obtains competitive results (only 2 score points drop).

The top entry lixx [35] uses a deeper convnet model (ImageNet pre-trained ResNet), a similar pixel-level tracking architecture, trains it over external segmentation data (using $\sim 120k$ pixel-level annotated images from MS-COCO and Pascal VOC for pre-training, and akin to [6] fine-tuning on the DAVIS₁₇ train and val sets, $\sim 10k$ annotated frames), and extends it with a box-level object detector (trained over MS-COCO and Pascal VOC, $\sim 500k$ bounding boxes) and a box-level object re-identification model trained over $\sim 60k$ box annotations (on both images and videos). We argue that

our system reaches comparable results with a significantly lower amount of training data.

Figure 11 provides qualitative results of LucidTracker on the test-dev set. The video results include successful handling of multiple objects, full and partial occlusions, distractors, small objects, and out-of-view scenarios.

Conclusion. We show that top results for multiple object tracking can be achieved via our approach that focuses on exploiting as much as possible the available annotation on the first video frame, rather than relying heavily on large external training data.

6.3 Ablation study

Table 13 explores in more details how the different ingredients contribute to our results.

We see that adding extra information (channels) to the system, either optical flow magnitude or semantic segmentation, or both, does provide 1 \sim 2 percent point improvement. The results show that leveraging semantic priors and motion information provides a complementary signal to RGB image and both ingredients contribute to the tracking results.

Combining in ensemble four different models ($f_{I+F+S} + f_{I+F} + f_{I+S} + f_I$) allows to enhance the results even further, bringing 3 percent point gain.

Method	DAVIS ₁₇ , test-dev set							
	Rank	Global mean \uparrow	Region, J			Boundary, F		
			Mean \uparrow	Recall \uparrow	Decay \downarrow	Mean \uparrow	Recall \uparrow	Decay \downarrow
sidc	10	45.8	43.9	51.5	34.3	47.8	53.6	36.9
YXLKJ	9	49.6	46.1	49.1	22.7	53.0	56.5	22.3
haamoon [56]	8	51.3	48.8	56.9	12.2	53.8	61.3	11.8
Fromandtozh [77]	7	55.2	52.4	58.4	18.1	57.9	66.1	20.0
ilanv [57]	6	55.8	51.9	55.7	17.6	59.8	65.8	18.9
voigtlaender [68]	5	56.5	53.4	57.8	19.9	59.6	65.4	19.0
lalalafine123	4	57.4	54.5	61.3	24.4	60.2	68.8	24.6
wangzhe	3	57.7	55.6	63.2	31.7	59.8	66.7	37.1
lixx [35]	2	66.1	64.4	73.5	24.5	67.8	75.6	27.1
LucidTracker	1	66.6	63.4	73.9	19.5	69.9	80.1	19.4

Table 11: Comparison of segment tracking results on DAVIS₁₇, test-dev set. Our LucidTracker shows top performance.

Method	DAVIS ₁₇ , test-challenge set							
	Rank	Global mean \uparrow	Region, J			Boundary, F		
			Mean \uparrow	Recall \uparrow	Decay \downarrow	Mean \uparrow	Recall \uparrow	Decay \downarrow
zwrq0	10	53.6	50.5	54.9	28.0	56.7	63.5	30.4
Fromandtozh [77]	9	53.9	50.7	54.9	32.5	57.1	63.2	33.7
wasidennis	8	54.8	51.6	56.3	26.8	57.9	64.8	28.8
YXLKJ	7	55.8	53.8	60.1	37.7	57.8	62.1	42.9
cjc [11]	6	56.9	53.6	59.5	25.3	60.2	67.9	27.6
lalalafine123	6	56.9	54.8	60.7	34.4	59.1	66.7	36.1
voigtlaender [68]	5	57.7	54.8	60.8	31.0	60.5	67.2	34.7
haamoon [56]	4	61.5	59.8	71.0	21.9	63.2	74.6	23.7
vantam299 [33]	3	63.8	61.5	68.6	17.1	66.2	79.0	17.6
LucidTracker	2	67.8	65.1	72.5	27.7	70.6	79.8	30.2
lixx [35]	1	69.9	67.9	74.6	22.5	71.9	79.1	24.1

Table 12: Comparison of segment tracking results on DAVIS₁₇, test-challenge set. Our LucidTracker shows competitive performance, holding the second place in the competition.

Our lucid dreams enable automatic CRF-tuning (see Section 5.3.3) which allows to further improve the results (65.2 \rightarrow 66.6 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. Notice that even without the semantic segmentation signal S_t our ensemble result is competitive.

6.4 Error analysis

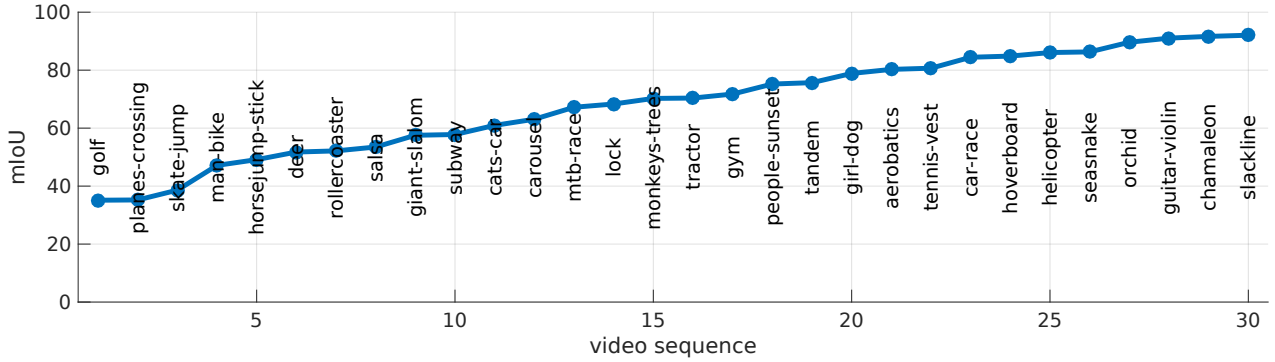
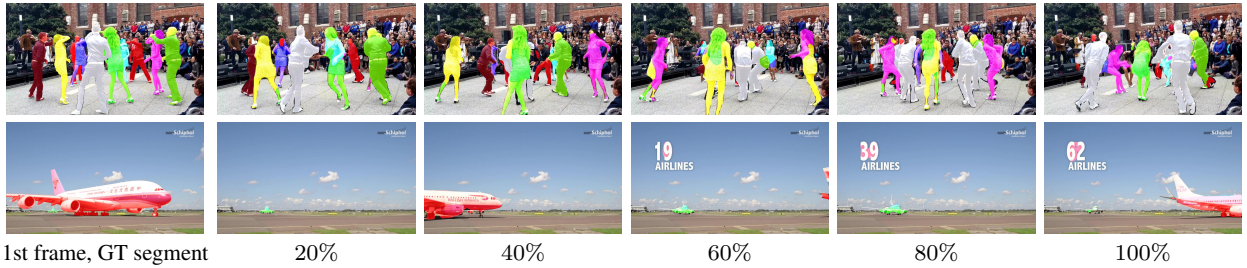
We present the per-sequence results of LucidTracker on DAVIS₁₇ in Figure 12 (per frame results not available from evaluation server). We observe that this dataset is significantly more challenging than DAVIS₁₆ (compare to Figure

10), with only $1/3$ of the test videos above 80 mIoU. This shows that multiple object tracking is a much more challenging task than tracking a single object.

The failure cases discussed in Section 5.5 still apply to the multiple objects case. Additionally, on DAVIS₁₇ we observe a clear failure case when tracking similar looking object instances, where the object appearance is not discriminative to correctly track the object, resulting in label switches or bleeding of the label to other look-alike objects. Figure 13 illustrates this case. This issue could be mitigated by using object level instance identification modules, like [35], or by changing the training loss of the model to more severely penalize identity switches.

Conclusion. In the multiple object case the LucidTracker results remain robust across different videos. The overall results being lower than for the single object tracking case,

Variant	\mathcal{I}	\mathcal{F}	\mathcal{S}	ensemble	CRF tuning	DAVIS ₁₇					
						test-dev			test-challenge		
						global mean	mIoU	mF	global mean	mIoU	mF
LucidTracker	✓	✓	✓	✓	✓	66.6	63.4	69.9	67.8	65.1	70.6
	✓	✓	✓	✓	✗	65.2	61.5	69.0	-	-	-
	✓	✓	✗	✓	✓	64.9	61.3	68.4	-	-	-
	✓	✓	✗	✓	✗	64.2	60.1	68.3	-	-	-
LucidTracker $\mathcal{I} + \mathcal{F} + \mathcal{S}$	✓	✓	✓	✗	✓	62.9	59.1	66.6	-	-	-
	✓	✓	✓	✗	✗	62.0	57.7	62.2	64.0	60.7	67.3
$\mathcal{I} + \mathcal{F}$	✓	✓	✗	✗	✗	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 13: Ablation study of different ingredients. DAVIS₁₇, test-dev and test challenge sets.Figure 12: Per-sequence results on DAVIS₁₇, test-dev set.Figure 13: LucidTracker failure cases on DAVIS₁₇, test-dev set. Frames sampled along the video duration (e.g. 50%: video middle point). We show 2 results mIoU over the video below 50.

there is more room for future improvement in the multiple object pixel-level tracking task.

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). Even more, 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.

Our extensive analysis decomposed the ingredients that contribute to our improved results, indicating that our new training strategy and the way we leverage additional cues such as semantic and motion priors are key.

Showing that training a convnet for object tracking can be done with only few (~ 100) training samples changes the mindset regarding how much general knowledge about ob-

jects is required to approach this problem [28, 26], 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 single and multiple object tracking.

Acknowledgements

Eddy Ilg and Thomas Brox acknowledge funding by the DFG Grant BR 3815/7-1.

References

1. Ç. Aytekin, E. C. Ozan, S. Kiranyaz, and M. Gabbouj. Visual saliency by extended quantum cuts. In *ICIP*, 2015. 8
2. A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. *arXiv:1702.06506*, 2017. 2
3. L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. *arXiv:1606.09549*, 2016. 1, 2, 5
4. F. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *PAMI*, 1989. 6
5. M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *ICCV*, 2009. 2, 5
6. 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, 5, 10, 11, 13, 14, 15, 16
7. J. Chang, D. Wei, and J. W. Fisher. A video representation using temporal superpixels. In *CVPR*, 2013. 2
8. L. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017. 15
9. 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. 3, 10, 11
10. W. Chen, H. Wang, Y. Li, H. Su, Z. Wang, C. Tu, D. Lischinski, D. Cohen-Or, and B. Chen. Synthesizing training images for boosting human 3d pose estimation. In *3D Vision (3DV)*, 2016. 3
11. J. Cheng, S. Liu, Y.-H. Tsai, W.-C. Hung, S. Gupta, J. Gu, J. Kautz, S. Wang, and M.-H. Yang. Learning to segment instances in videos with spatial propagation network. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. 17
12. A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *Trans. Img. Proc.*, 2004. 6
13. M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCV workshop*, 2015. 2
14. A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 3
15. M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 4
16. A. Faktor and M. Irani. Video segmentation by non-local consensus voting. In *BMVC*, 2014. 2, 10, 14
17. G. Georgakis, A. Mousavian, A. C. Berg, and J. Kosecka. Synthesizing training data for object detection in indoor scenes. *arXiv:1702.07836*, 2017. 3
18. X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010. 8
19. M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010. 2
20. A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. In *CVPR*, 2016. 3
21. D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *ECCV*, 2016. 1, 2, 5
22. J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012. 2, 3, 5
23. 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, 8, 10, 11
24. S. D. Jain and K. Grauman. Supervoxel-consistent foreground propagation in video. In *ECCV*, 2014. 6
25. S. D. Jain and K. Grauman. Click carving: Segmenting objects in video with point clicks. In *HCOMP*, 2016. 2
26. S. D. Jain, B. Xiong, and K. Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. *arXiv:1701.05384*, 2017. 2, 3, 10, 14, 19
27. V. Jampani, R. Gadde, and P. V. Gehler. Video propagation networks. *arXiv:1612.05478*, 2016. 1, 2, 3, 10, 14
28. 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, 5, 10, 11, 13, 14, 15, 19
29. P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*. 2011. 5, 8, 11
30. M. Kristan, J. Matas, et al. The visual object tracking vot2015 challenge results. In *ICCV workshop*, 2015. 1
31. M. Kristan, J. Matas, et al. The visual object tracking vot2016 challenge results. In *ECCV workshop*, 2016. 1
32. M. Kristan, R. Plugfelder, et al. The visual object tracking vot2014 challenge results. In *ECCV workshop*, 2014. 1, 2, 5
33. T.-N. Le, K.-T. Nguyen, M.-H. Nguyen-Phan, T.-V. Ton, T.-A. N. (2), X.-S. Trinh, Q.-H. Dinh, V.-T. Nguyen, A.-D. Duong, A. Sugimoto, T. V. Nguyen, and M.-T. Tran. Instance re-identification flow for video object segmentation. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. 17
34. F. Li, T. Kim, A. Humayun, D. Tsai, and J. M. Rehg. Video segmentation by tracking many figure-ground segments. In *ICCV*, 2013. 1, 6, 8
35. X. Li, Y. Qi, Z. Wang, K. Chen, Z. Liu, J. Shi, P. Luo, C. C. Loy, and X. Tang. Video object segmentation with re-identification. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. 16, 17
36. G. Lin, A. Milan, C. Shen, and I. D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *arXiv:1611.06612*, 2016. 2
37. C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015. 2
38. N. Maerki, F. Perazzi, O. Wang, and A. Sorkine-Hornung. Bilateral space video segmentation. In *CVPR*, 2016. 2, 5, 10, 14, 15
39. N. Nagaraja, F. Schmidt, and T. Brox. Video segmentation with just a few strokes. In *ICCV*, 2015. 2, 10
40. H. Nam, M. Baek, and B. Han. Modeling and propagating cnns in a tree structure for visual tracking. *arXiv:1608.07242*, 2016. 2
41. H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 1, 2
42. 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. 3
43. A. Papazoglou and V. Ferrari. Fast object segmentation in unconstrained video. In *ICCV*, 2013. 2
44. D. Park and D. Ramanan. Articulated pose estimation with tiny synthetic videos. In *CVPR Workshop*, 2015. 3

45. D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. *arXiv:1612.06370*, 2016. **5**
46. 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, 6, 13**
47. F. Perazzi, O. Wang, M. Gross, and A. Sorkine-Hornung. Fully connected object proposals for video segmentation. In *ICCV*, 2015. **2, 10, 14**
48. L. Pishchulin, A. Jain, M. Andriluka, T. Thormählen, and B. Schiele. Articulated people detection and pose estimation: Reshaping the future. In *CVPR*, 2012. **3**
49. T. Pohlen, A. Hermans, M. Mathias, and B. Leibe. Full-resolution residual networks for semantic segmentation in street scenes. In *CVPR*, 2017. **15**
50. J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. Davis challenge on video object segmentation 2017. <http://davischallenge.org/challenge2017>. **16**
51. 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. **15**
52. A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012. **1, 6**
53. J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epiflow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, 2015. **11**
54. S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. **3**
55. C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *SIGGRAPH*, 2004. **3, 8**
56. A. Shaban, A. Firl, A. Humayun, J. Yuan, X. Wang, P. Lei, N. Dhandha, B. Boots, J. M. Rehg, and F. Li. Multiple-instance video segmentation with sequence-specific object proposals. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. **17**
57. G. Sharir, E. Smolyansky, and I. Friedman. Video object segmentation using tracked object proposals. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. **17**
58. K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. **8, 15**
59. T. V. Spina and A. X. Falcão. Fomtrace: Interactive video segmentation by image graphs and fuzzy object models. *arXiv:1606.03369*, 2016. **2**
60. J. Sun, J. Jia, C.-K. Tang, and H.-Y. Shum. Poisson matting. In *SIGGRAPH*, 2004. **6**
61. K. Tang, V. Ramanathan, L. Fei-fei, and D. Koller. Shifting weights: Adapting object detectors from image to video. In *NIPS*, 2012. **5**
62. M. Tang, D. Marin, I. Ben Ayed, and Y. Boykov. Normalized cut meets mrf. In *ECCV*, 2016. **3**
63. S. Tang, M. Andriluka, A. Milan, K. Schindler, S. Roth, and B. Schiele. Learning people detectors for tracking in crowded scenes. In *ICCV*, 2013. **3**
64. R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. *arXiv:1605.05863*, 2016. **2**
65. P. Tokmakov, K. Alahari, and C. Schmid. Learning motion patterns in videos. *arXiv:1612.07217*, 2016. **2, 10, 14**
66. Y.-H. Tsai, M.-H. Yang, and M. J. Black. Video segmentation via object flow. In *CVPR*, 2016. **2, 10, 14, 15**
67. G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid. Learning from synthetic humans. *arXiv:1701.01370*. **3**
68. P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for the 2017 davis challenge on video object segmentation. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. **17**
69. T. Vojir and J. Matas. Pixel-wise object segmentations for the VOT 2016 dataset. Research report, 2017. **1**
70. L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2015. **2**
71. T. Wang, B. Han, and J. Collomosse. Touchcut: Fast image and video segmentation using single-touch interaction. *CVIU*, 2014. **2**
72. W. Wang and J. Shen. Super-trajectory for video segmentation. *arXiv:1702.08634*, 2017. **2, 5, 10, 14**
73. X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. **5**
74. Z. Wu, C. Shen, and A. van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *arXiv:1611.10080*, 2016. **2**
75. F. Xiao and Y. J. Lee. Track and segment: An iterative unsupervised approach for video object proposals. In *CVPR*, 2016. **2, 10**
76. J. J. Yu, A. W. Harley, and K. G. Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. *arXiv:1608.05842*, 2016. **5**
77. H. Zhao. Some promising ideas about multi-instance video segmentation. *The 2017 DAVIS Challenge on Video Object Segmentation - CVPR Workshops*, 2017. **17**
78. H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. **2, 4**
79. Y. Zhu, Z. Lan, S. Newsam, and A. G. Hauptmann. Guided optical flow learning. *arXiv:1702.02295*, 2017. **5**