Universität des Saarlandes
Max-Planck-Institut für Informatik

Learning to Track Humans in Videos

Master's Thesis in Computer Science
by

Mihai Fieraru

supervised by

Prof. Dr. Bernt Schiele

advised by

MSc Anna Khoreva

MSc Eldar Insafutdinov

reviewers

Prof. Dr. Bernt Schiele

Dr. Mario Fritz

Saarbrücken, December 2017

Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, December 2017 Mihai Fieraru

Abstract

This thesis addresses the multi-person tracking task with two types of representation: body pose and segmentation mask. We explore these scenarios in the semi-supervised setting, where one available annotation is available per person during test time. More complex representations of people (segmentation mask and body pose) can provide richer understanding of visual scenes, and methods that leverage supervision during test time should be developed for the cases when supervision is available.

We propose `HumanMaskTracker` for the task of semi-supervised multi-person mask tracking. Our approach builds on recent techniques proposed for the task of video object segmentation. These include the mask refinement approach, training with synthetic data, fine-tuning per object and leveraging optical flow. In addition, we propose leveraging instance semantic segmentation proposals to give the tracker a better notion about the human class. Moreover, we propose modeling people occlusions inside the data synthesis process to make the tracker more robust to the challenges of occlusion and disocclusion.

For the task of semi-supervised multi-person pose tracking, we propose the method `HumanPoseTracker`. We show that the task of multi-person pose tracking can benefit significantly from using one pose supervision per track during test time. Fine-tuning per object and leveraging optical flow, techniques proposed for the task of video object segmentation, prove to be highly effective for supervised pose tracking as well. Also, we propose a technique to remove false positive joint detections and develop tracking stopping criteria. A promising application of our work is presented by extending the method to generate dense from sparse annotations in videos.

Acknowledgements

I would like to express my sincere gratitude to Prof. Dr. Bernt Schiele for taking me aboard on his group and supervising me on this enjoying research project. Through this I could meet Anna Khoreva and Eldar Insafutdinov, to whom I deeply thank for their guidance, both professional and technical, as well as for providing advice for my presentations and proofreading this thesis. I am grateful to Dr. Mario Fritz for dedicating the time to be the second reviewer of this work.

I would like to thank all members of the Computer Vision and Multimodal Computing department for creating a friendly and intellectually nourishing environment that has made me cherish my time at the Max Planck Institute for Informatics.

I am thankful to the International Max Planck Research School for Computer Science for providing the generous facilities as well as a scholarship to fund my master studies at Saarland University and allowing me to meet the friends I now have.

Finally, I am thankful to my family for their support in all dimensions. Without them, none of this would have been possible.

Contents

Chapter 1

Introduction

1.1 Motivation

Enabling computers to understand visual scenes containing people has been a long researched problem in the computer vision community. The interest in this topic is driven by several reasons.

First, understanding scenes with people can be seen as a necessary step towards solving the more general visual scene understanding task. Not only have visual scenes a tendency to contain humans, but also people are one of the most complex object class. Thus, the challenges and importance of the human category motivate addressing this class separately.

Second, various applications can be enabled by advances in this field. Improving driver-assistance systems could make traffic safer for pedestrians. Robots that could understand people's actions would help us with physical tasks. Human pose understanding could be used in the gaming industry and people tracking could be used in surveillance systems.

There are various ways of representing instances of people in images: bounding box, human keypoints (pose), segmentation masks, and many others. Estimating these representations can also be done in image sequences (called tracking), with the added requirement of maintaining identities of people over time.

In this thesis, we address tracking of two representations of people: their 2D body pose and their segmentation mask. We consider the unconstrained scenario in which multiple people can be present in the scene. Tracking of people is important for its wide range of applications, from video surveillance to activity recognition and behavior

understanding. The motivation behind tracking body poses and segmentation masks of people is that they are more complex representations than the usual bounding boxes, hence they can enable a richer understanding of scenes.

Based on the levels of supervision assumed during testing, tracking can be categorized into unsupervised tracking, semi-supervised tracking (generally requiring one or more annotated frames) and supervised tracking (requiring some type of annotation in each frame).

In this work, we propose the task of semi-supervised people tracking, which, to the best of our knowledge, has not been addressed before. One motivation for developing techniques that assume semi-supervision is that, in the case semi-supervision is available, then it should be leveraged to improve tracking of people. Another motivation for this scenario would be to use tracking to propagate people annotations to the neighboring frames and reduce annotations costs.

In more detail, we address the following tasks:

- ***Semi-Supervised Multi-Person Segmentation Tracking*** (also referred to in this thesis as *Video Human Segmentation*, *Human Segmentation Tracking*, or *Human Mask Tracking*).
  The task assumes supervision in the first frame of the video sequence, in the form of one mask segmentation for each person that has to be tracked. The task is to correctly predict in all consequent frames the segmentation mask of each of these first frame annotated persons.

- ***Semi-Supervised Multi-Person Pose Tracking*** (also referred to in this thesis as *Multi-Person Pose Tracking*, *Human Pose Tracking*, or just *Pose Tracking*).
  The task assumes one supervision from each person to be tracked in the video. The ground truth (GT) pose is collected from the middle of the track of the target person. The task is to correctly estimate the body pose of each of the persons to be tracked in all frames of the video,

We treat the two tasks separately, although we think they could benefit from one another in a joint setting.

For their effectiveness at learning the appearance of objects, we research convolutional neural network (CNN) models for both tracking tasks.

1.2 Challenges

The two tasks that we address come with similar challenges.

First, as in any learning system, the challenges of people tracking are stemming, among others, by the data available for learning. Annotating data is expensive and, in the case of videos, it manifests via a trade-off between datasets with few long sequences and datasets with many short sequences (or just many static images). This comes at the price of either limited appearance variability or limited motion information. Several implied limitations:

- the development of end-to-end learning methods (that directly predict pose tracks from video) is hard to achieve; end-to-end methods require larger amounts of annotated data;
- using temporal information for learning stays challenging; recurrent neural networks (RNNs) add a new dimension in the input space; this highly increases the number of parameters, and implicitly the required data for learning them;

Second, the task of tracking comes with its particular challenges:

- human motions and change of camera viewpoints alter the appearance, shape and size of people across time;
- fast motion and/or occlusions break the temporal consistency assumption (the changes in appearance, shape, size from one frame to another are small);
- similar looking people (usually in sport scenes where clothing is almost identical) are hard to track even for people;
- people can disappear and reappear in the scene (when they move or when the scene viewpoint changes);

1.3 Contributions

For the two tasks that we introduce, we present the following contributions:

Human Mask Tracking

- we adopt techniques proposed for the task of video object segmentation to the task of multi-person mask tracking;

- we make the method more robust to occlusions and disocclusions;
- we leverage information from an image-level instance semantic segmentation method by learning about the human class;

Human Pose Tracking

- we adopt techniques proposed for the task of video object segmentation to the task of multi-person pose tracking;
- we develop criteria to curate keypoint detections;
- we show that the method leverages the supervision to perform better tracking than the state of the art unsupervised pose tracker;
- we show that the method can be used to generate dense from sparse annotations;

Chapter 2

Related Work

Our proposed methods are related to previous works involving image-level segmentation and pose estimation, as well as video-level object tracking and pose tracking.

2.1 Image Level

Naturally, many of the recent advances in tracking have been preceded by and linked to advances on image-level tasks.

2.1.1 Segmentation

Semantic Segmentation is the task of classifying each pixel of an image to a semantic class. In this field, the development of fully convolutional networks (FCNs) by [Long et al., 2015] has proved particularly influential, opening the door for end-to-end learning for semantic segmentation. By turning fully connected layers of a CNN into convolutional layers, [Long et al., 2015] transformed the CNN network from doing one dimensional classification to performing pixel level classification. One particular problem with this approach is the loss of spacial resolution due to use of strides in CNNs. Recent approaches proposed different solutions for this issue, from using skip-layer connections [Chen et al., 2016], to using dilated convolution [Chen et al., 2016] or using encoder-decoder networks [Noh et al., 2015]. Also, post-processing with CRFs [Krähenbühl and Koltun, 2011] is another technique used for smoothing the segmentation output.

In our Human Mask Tracking method, we repurpose the DeepLabv2 [Chen et al., 2016] architecture designed for semantic segmentation to predict an instance segmentation frame by frame.

Semantic Instance Segmentation is the task that in addition to classifying each pixel in an image to a semantic class, it also groups pixels into object instances. A recent approach combines object detection with semantic segmentation into fully convolutional instance segmentation (FCIS) [Yi Li and Wei, 2017]. The current state of the art is Mask-RCNN [He et al., 2017], which extends Faster-RCNN [Ren et al., 2015] by adding one more branch for predicting segmentation masks in each region of interest.

In our experiments, we use FCIS, as the implementation of their method is publicly available.

2.1.2 Pose Estimation

Pose Estimation also benefited significantly from the development of fully convolutional networks. Most approaches treat each keypoint as a segmentation mask. Different keypoint types can overlap with each other, case which is not punished during training. In the case of single-person pose estimation, detection of keypoints is done by locating the pixel with the highest confidence in each keypoint output channel. In the case of multi-person pose estimation, there are two types of approaches. Bottom-up methods [Insafutdinov et al., 2016], [Cao et al., 2016] first detect keypoints and then group them together into people. Top-down approaches [He et al., 2017], [Papandreou et al., 2017], on the other hand, first detect human bounding boxes and then apply single-person pose estimation within each box.

In our experiments, we build on top of DeeperCut [Insafutdinov et al., 2016], which we use as a single-person pose estimator.

2.2 Video Level

2.2.1 Object Tracking

Object tracking is a long studied problem in the computer vision field. If, originally, object tracking was identified with tracking of boxes, now the term can also refer to mask tracking.

2.2.1.1 Box Tracking

The classic box tracking task is semi-supervised. The bounding box of the object to be tracked is provided in the first frame and the task is class-agnostic. While there is a large

body of work addressing this task, our method is related to recent approaches based on CNNs. [Held et al., 2016] proposes training a CNN to regress the bounding box frame by frame, based on the object's appearance and its previous frame prediction. [Nam et al., 2016] proposed to fine tune the CNN tracker with object's appearance. Both techniques are adopted by [Khoreva et al., 2017b] and showed to be effective in class agnostic mask tracking. We also adopt them for tracking masks and poses of people.

There is some body of work also addressing tracking of people as a task on its own. The approaches are unsupervised and cover mostly tracking of people in street scenes [Andriluka et al., 2008], [Milan et al., 2014], [Tang et al., 2017]. We are addressing a more general scenario, where people appear in all types of environments and their poses are not restricted to the standing or walking position.

2.2.1.2 Mask Tracking

Mask Tracking, also called Video Object Segmentation, is the task of tracking the mask segmentation of an object through a video. The task is semi-supervised, receiving the mask of the target object in the first frame of the video. Recent methods adopt the previously mentioned box tracking techniques while repurposing semantic segmentation architectures to obtain mask predictions.

[Khoreva et al., 2017a] proposes to synthesize possible future frames by exploiting the supervision in the first frame. This is shown to fight the problems of domain shift between training set and the testing set, as well as the dependence on large scale segmentation datasets. Our Human Mask Tracking method also builds on this technique and extends it to the multi-person scenario.

2.2.2 Multi-Person Pose Tracking

The multi-person pose tracking task was recently proposed in [Insafutdinov et al., 2017] and [Iqbal et al., 2017], and involves tracking of poses of multiple people in unrestricted scenes, without any supervision available during test time. The two initial approaches propose a bottom-up approach. First, they detect all keypoints of all people in all frames of the video. Then, an integer program optimization is grouping keypoints into people over time.

More recent techniques are proposed in the PoseTrack challenge [Andriluka et al., 2017]. They all approach the problem following a tracking-by-detection framework. First, multi-person pose estimation is performed in each frame of the video. Second, simple matching algorithms are used to link the detections over time.

The pose tracking task we are targeting in this work is different, as it assumes supervision during test time.

Chapter 3

Human Segmentation Tracking

3.1 Method

In this section, we detail our approach on the Human Segmentation Tracking task. We assume the tracker receives supervision in the first frame of the video.

3.1.1 Architecture

Due to recent advances in Video Object Segmentation, we build upon the work of MaskTrack [Khoreva et al., 2017b]. To extend the approach to multiple people, we address tracking each person in the video as a separate problem.

MaskTrack approaches the problem of segmentation tracking as a mask refinement task. To segment a particular instance, the prediction is guided by the appearance of the object (the RGB image), temporal consistency (the previous frame mask) and motion information (the optical flow). We use the same technique to track masks of humans. In addition, in order to make the tracker people oriented, we employ human-specific information from an image-level semantic instance segmentation system.

Formally, we predict the mask M_t^h of a human h in frame t as:

$$M_t^h = f(I_t, w(M_{t-1}^h, \mathcal{F}_t), \mathcal{S}_t^h) \tag{3.1}$$

where:

- f is the `HumanMaskTracker`, the function that we want to learn and use for frame-by-frame prediction. Depending on the training procedure, the function can be adapted *per-person* via fine-tuning (see Section 3.1.2.3).

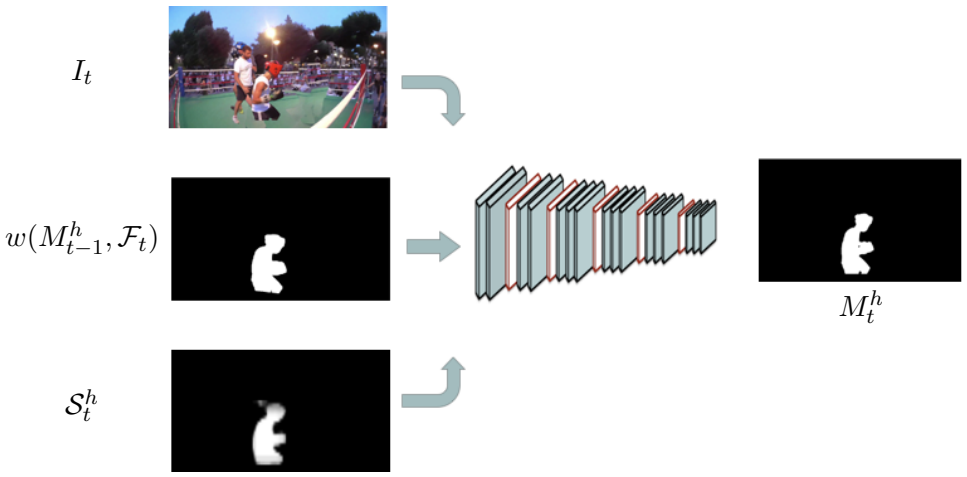


FIGURE 3.1: `HumanMaskTracker` architecture. Inputs include the RGB image I_t, the previous frame mask and motion information as $w(M^h_{t-1}, \mathcal{F}_t)$, and semantic instance segmentation information $\mathcal{S}^h_t$. The tracker is trained to predict M^h_t.

We model the function using a fully convolutional neural network designed for semantic segmentation, DeepLabv2 [Chen et al., 2016], based on the VGG [Simonyan and Zisserman, 2014] architecture. We also experimented with the ResNet [He et al., 2016] version of DeepLabv2, but the increase in performance was outweighed by the longer training time. Although semantic segmentation networks usually accept only RGB images as input, they can be easily extended to accept additional channels by increasing the depth of the filters of the first convolutional layer (from 3 to $3 + n$, where n is number of extra channels).

- I_t is the **Current Image** at frame t, in RGB color mode
- M^h_{t-1} is the **Previous Frame Mask** of person h predicted on image I_{t-1}. In the case where $t = 1$, the previous frame mask M^h_0 is already available as ground truth due to the semi-supervised setting of the task.

  The motivation to condition the prediction of M^h_t by M^h_{t-1} comes from the temporal consistency assumption, i.e. objects do not change their shape and position too much from one frame to another. Therefore, the previous predicted mask can be a good indicator of the location and shape of the mask in frame t.
- $\mathcal{F}_t$ is the **Optical Flow** between image I_{t-1} and image I_t, which estimates the motion vector between the two frames. In our case, we compute it using FlowNet2.0 [Ilg et al., 2017], which is itself a trained convolutional neural network. It is estimated by only using the two RGB images I_{t-1} and I_t as input.
- w is an operation which **warps** a binary mask with the optical flow. We warp the previous frame mask M^h_{t-1} with the estimated optical flow $\mathcal{F}_t$ in order to have a better localized mask. While M^h_{t-1} is a good estimate of M^h_t, the warped version

$w(M^h_{t-1}, \mathcal{F}_t)$ is even more indicative, as it incorporates motion information. This method of leveraging optical flow was previously shown to be effective in [Khoreva et al., 2017a].

- $\mathcal{S}^h_t$ is the **Semantic Instance Segmentation** of person h in image I_t. The motivation behind employing semantic instance segmentation is to guide the mask propagation with specific information learned about the human class. We also experimented with semantic segmentation information (no instances), but leveraging information about instances of people is more effective.

  As shown in Figure 3.2, semantic instance segmentation outputs multiple mask instances of different classes. To provide the network with an additional cue about the location and shape of the human we are tracking, we select only the instance of the person which overlaps the most with $w(M^h_{t-1}, \mathcal{F}_t)$. If such an instance exists and the overlap is not 0, $\mathcal{S}^h_t$ will be the confidence map corresponding to that mask, for it contains richer information than the binary mask. Otherwise, $\mathcal{S}^h_t$ will be the null matrix. By confidence map, we refer to the CNN output where the softmax segmentation loss is applied during training. It can be seen as a matrix indicating the probability of each image pixel to be part of the foreground mask. The overlap of the two masks is computed using the Jaccard Index (see Equation 3.2). The selection process is illustrated in Figure 3.3

  To compute the semantic instance segmentation, we use the FCIS network [Yi Li and Wei, 2017], which won the first place in the COCO segmentation challenge 2016. It is trained on COCO [Lin et al., 2014] to predict instance masks of 80 categories (with person being the most represented category).

See Figure 3.1 for a visualization of the simplified `HumanMaskTracker` architecture.

Once trained, the `HumanMaskTracker` is initialized with the first frame ground-truth mask of the person it is supposed to track and then proceeds with predicting the future masks frame by frame, following the temporal order. Note that the tracker is not fine-tuned from one frame to another, only its input changes.

3.1.2 Training Stage

The fact that the tracker architecture does not exploit longer temporal information (over many frames) comes to the benefit of the training procedure, which does not require expensive densely annotated frames. Instead of using ground-truth masks M^h_{t-1}, M^h_t of consecutive frames, [Khoreva et al., 2017b] proposes using only the ground-truth mask M^h_t for the frame I_t and instead synthesize $w(M^h_{t-1}, \mathcal{F}_t)$. This way, the network can be

FIGURE 3.2: FCIS output masks overlaid onto the image. Notice multiple object types (person, car, bus) and multiple instances of the same object type (yellow and purple persons).

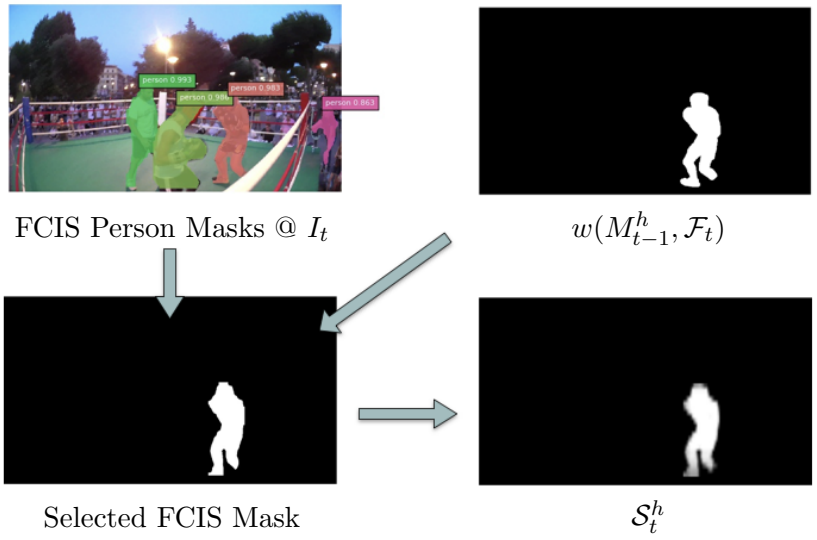


FIGURE 3.3: Process of selecting $\mathcal{S}^h_t$. We start from all FCIS person masks (upper left corner shows 4 people detected) and compute their overlap with $w(M^h_{t-1}, \mathcal{F}_t)$ (upper right corner). The mask with the highest overlap is selected (lower left corner) and $\mathcal{S}^h_t$ will be its corresponding confidence map (lower right corner).

trained with static images only, using image segmentation datasets. Our work adopts Lucid Data Dreaming [Khoreva et al., 2017a], which takes the idea of data generation even further by also synthesizing image data I^h_t. This is very beneficial as it addresses the problem of domain shift between training and testing data.

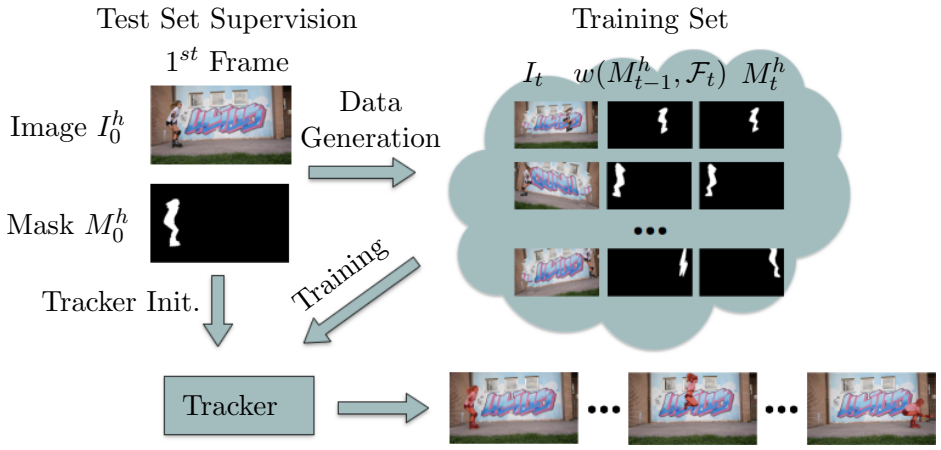


FIGURE 3.4: Lucid data dreaming overview. To track person h, LucidDream first uses the ground truth M^h_0 of I^h_0 to synthesize possible future frames and annotations. It then trains the tracker with the generated data, making the tracker adapt to the particular sequence and person.

3.1.2.1 Lucid Data Dreaming

Lucid Data Dreaming addresses two important challenges of tracking with CNNs: on one hand, the need of large datasets for training convolutional neural networks and, on the other hand, the problem of domain shift between training and test data.

To ensure that the tracker is trained with samples close to the test sequence, the work proposes to use the annotated first frame of the test sequence to synthesize data specific to that sequence. The idea is to simulate possible future frames of the video, train the network using them and have the tracker adapt better to the upcoming frames, as illustrated in Figure 3.4.

To simulate future changes that can occur to the tracked object and the background, [Khoreva et al., 2017a] proposes to use the mask M^h_0 to cut-out the object, inpaint the background in that area, apply deformations to the cut-out object and then merge the new object onto the new background. More specifically, the following operations are applied:

- **Foreground** (Fig. 3.5D)**/ Background** (Fig. 3.5C) **split**: the mask M^h_0 (Fig. 3.5B) indicates the pixels of the foreground object and removes them from the image I_0 (Fig. 3.5A).
- **Background inpainting** (Fig. 3.5E): fill in the empty pixels in the background by inpainting using [Criminisi et al., 2004].

- **Object appearance change** (Fig. 3.5F): apply the following transformations to the foreground: random rotation $\pm 30°$, random scaling $\pm 15\%$ and random thin-plate splines deformations of $\pm 10\%$ of the object size [Bookstein, 1989].
- **Object location change** (Fig. 3.5F): randomly place the object within the image boundaries.
- **Foreground / Background merge** (Fig. 3.5G): applying Poisson matting to remove merging artifacts.
- **Illumination changes**: modify the H and V channels in the HSV color encoding with the function: $f(x) = ax^b + c$, where $a \in 1 \pm 0.05$, $b \in 1 \pm 0.3$, $c \in \pm 0.07$.
- **Camera motion**: apply the previously explained random affine transformations to the new image.

All these transformations operate on I_0 and create a synthetic image I_t. As the parameters of these operations are known, the ground-truth mask annotation M_t^h can also be computed (Fig. 3.5H).

Lucid Data Dreaming also synthesizes $w(M_{t-1}^h, \mathcal{F}_t)$. The warped previous frame mask $w(M_{t-1}^h, \mathcal{F}_t)$ is, in essence, a noisy version of M_t^h, with noise introduced by the prediction in the previous frame and the estimation of the optical flow. Instead of synthesizing M_{t-1}^h and $\mathcal{F}_t$ separately, [Khoreva et al., 2017a] synthesizes $w(M_{t-1}^h, \mathcal{F}_t)$ directly (Fig. 3.5I) by applying the previously explained random affine transformations and the random thin-plate splines deformations to M_t^h. This simulates the noise that we expect from $w(M_{t-1}^h, \mathcal{F}_t)$ at test time.

The synthesis process described above can generate a large set of annotated training samples. In their experiments, [Khoreva et al., 2017a] generate around 2.5k samples for each object to be tracked.

3.1.2.2 Lucid Data Dreaming for Multiple People

Originally designed for the class-agnostic single object tracking, Lucid Data Dreaming can be better tailored for the multiple humans tracking scenario.

Firstly, humans tend to interact with each other a lot. From the video sequence perspective, this translates to many occlusions being present, which pose difficulties to trackers. In Lucid Data Dreaming, the tracked person is always fully visible (see Figure 3.5G), so the tracker does not learn to recognize the person when it is occluded.

To this end, we propose synthesizing data jointly for all people in the sequence, which allows to synthesize people-to-people occlusion (see Figure 3.6F).

Secondly, people generally stand on the ground, so video sequences are more probable to show people touching the ground in the lower half of the image. Lucid Data Dreaming places the objects randomly and uniformly within the image boundaries, which creates many cases of the "flying people" artifact (see Figure 3.5G). To mitigate this issue, we propose placing people under an imaginary horizon line (see Figure 3.6F).

In more details, the following operations are applied:

- **Foreground / Background** (see Fig. 3.6B) **split** - cut-out all annotated humans h in the sequence.
- **Background inpainting** (see Fig. 3.6C)
- **Change appearance of each human** h
- **Change each person's location**:
  - randomly place on the x axis such that there is a 50% chance of occlusion with another person (any type of overlap of the bounding boxes on the x axis); we also enable truncations (occlusions by the image boundaries);
  - randomly place on the y axis such that the bottom of the person bounding box is located under an imaginary horizon line in the middle of the y axis.
- **Foreground / Background merge** - blend one person at a time (See Fig. 3.6D, Fig. 3.6E and Fig. 3.6F)
- **Illumination changes**
- **Camera motion**

These transformations create a synthetic image I_t (Fig. 3.6F) and one ground-truth mask M_t^h for each human h annotated in the sequence (see all masks in Fig. 3.6G and a particular one in Fig. 3.6I). We then apply the same previously explained non-rigid and affine transformations to each non-occluded mask M_t^h to simulate the noise expected at test time. After this is done independently for each person, we merge the masks in the same order as in the merging of the foregrounds images. This gives us the $w(M_{t-1}^h, \mathcal{F}_t)$ for each human h annotated in the sequence (see all masks $w(M_{t-1}^h, \mathcal{F}_t)$ in Fig. 3.6H and a particular one in Fig. 3.6J).

Unless stated otherwise, the transformations follow the same choice of parameters as [Khoreva et al., 2017a], as explained in Section 3.1.2.1. Figures 3.6I and 3.6J show

that in this case of data synthesis, the tracked persons are not always fully visible, hence some occlusions scenarios are modeled.

Note that the semantic instance segmentation channel also has to be generated. While at test time we were running the FCIS network over the test image, at training time we run it over the synthetic image. To indicate to the network which person we are tracking, we select only the instance of the person closest to M_t^h. If such an instance exists, $\mathcal{S}_t^h$ will be the confidence map corresponding to that mask. Otherwise, $\mathcal{S}_t^h$ will be the null matrix. As described before, at test time we instead compute the overlap with $w(M_{t-1}^h, \mathcal{F}_t)$ as an estimate of M_t^h.

With the synthesis method described above, we generate around 3.5k samples for each tracked person.

3.1.2.3 Training Modalities

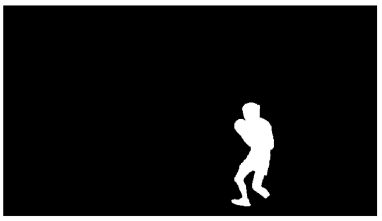
We adopt the training procedure that produces the best tracking results in [Khoreva et al., 2017a]. This includes the following stages, in this order:

1. **ImageNet pre-training** - a common training stage in most CNNs; pre-training on a large classification dataset like ImageNet [Deng et al., 2009] ensures that the network is properly learning generic feature extractors that it might otherwise not be able to learn only with a smaller dataset;

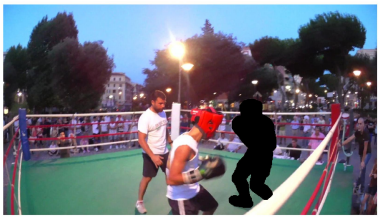
2. **Per-dataset training** - uses the synthetic data generated for all test persons in the test set; its purpose is to train the network for the target task of mask tracking; in our case, it learns how masks of persons look in general, what pixel represents a human, what pixel does not;

3. **Per-person tuning** - uses the synthetic data generated for a specific test person to train the network used to track solely that person; this creates one tracker f_h per test person h, which is able to recognize the appearance of the person it was trained with;

(A) I_0



(B) M_0^h



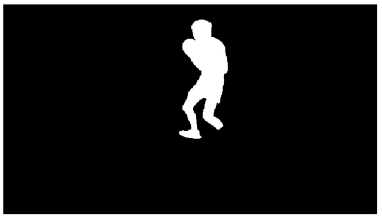
(C) Background

(D) Foreground

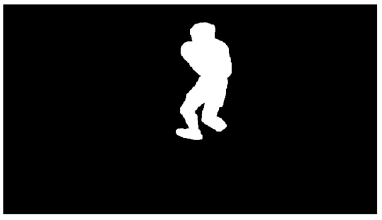
(E) Inpainting

(F) Appearance and location change

(G) I_t



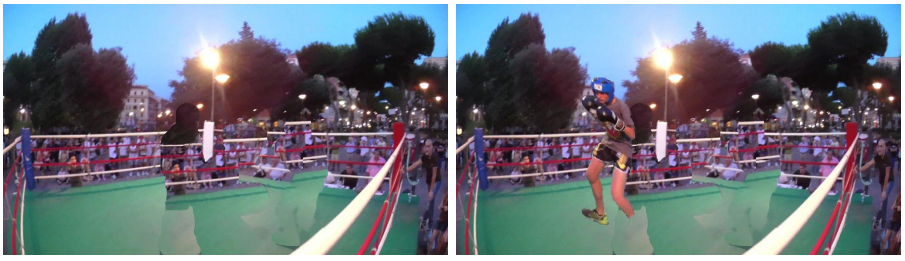
(H) M_t^h



(I) $w(M_{t-1}^h, \mathcal{F}_t)$

FIGURE 3.5: Example transformations applied by LucidDream. Note that, for brevity, the illumination changes and the camera motion operations are not shown.

(A) I_0 (B) Background



(C) Inpainting (D) Merge first person

(E) Merge second person (F) Merge third person - I_t

(G) $M_t^h, h \in \{h_1, h_2, h_3\}$ (H) $w(M_{t-1}^h, \mathcal{F}_t), h \in \{h_1, h_2, h_3\}$

(I) $M_t^{h_1}$ (J) $w(M_{t-1}^{h_1}, \mathcal{F}_t)$

FIGURE 3.6: Example transformations applied by Lucid Data Dreaming for Multiple People. Note that, for brevity, the illumination changes and the camera motion operations are not shown.

3.2 Experiments

3.2.1 Experimental Setup

3.2.1.1 Dataset

To the best of our knowledge, there is no available dataset containing dense mask annotations of people. Nonetheless, the DAVIS 2017 dataset presented in [Pont-Tuset et al., 2017] contains videos densely annotated with pixel-level masks of object instances (see Figure 3.7). Although the dataset is class agnostic, it can be seen that many of the objects which are annotated are actually humans. Therefore, we manually select the sequences containing at least one person and remove all the non-person object annotations. We call this subset of annotations `DAVIS Persons`.

Figure 3.8 shows ground truth annotations of the `DAVIS Persons` dataset. The new dataset contains 58 sequences with a total of 93 people. 20 sequences contain more than one person per sequence. The average length of the sequences is ≈ 70 frames.

As in Davis 2017, all annotated people are present in the sequence starting at the first frame. However, not all people that are visible in the first frame are annotated. The ones that are not annotated tend to be located in the background and have smaller sizes.

The scenes are complex and present diverse challenges:

- *Appearance Change*: people's appearance changes during the video
- *Fast Motion* and *Motion Blur*: shown in [Perazzi et al., 2016] to be mutually dependable
- *Occlusions* and *Truncations*: people's masks partially or fully disappear due to occlusions with other people, objects or the image boundaries; the opposite event, *Disocclusion*, when the mask reappears, is also present
- *Scale Variation*: people's sizes vary during the video
- *Similar Looking People*: people looking alike due to similar clothing
- *Abrupt Camera Motion*: sudden movement of the camera

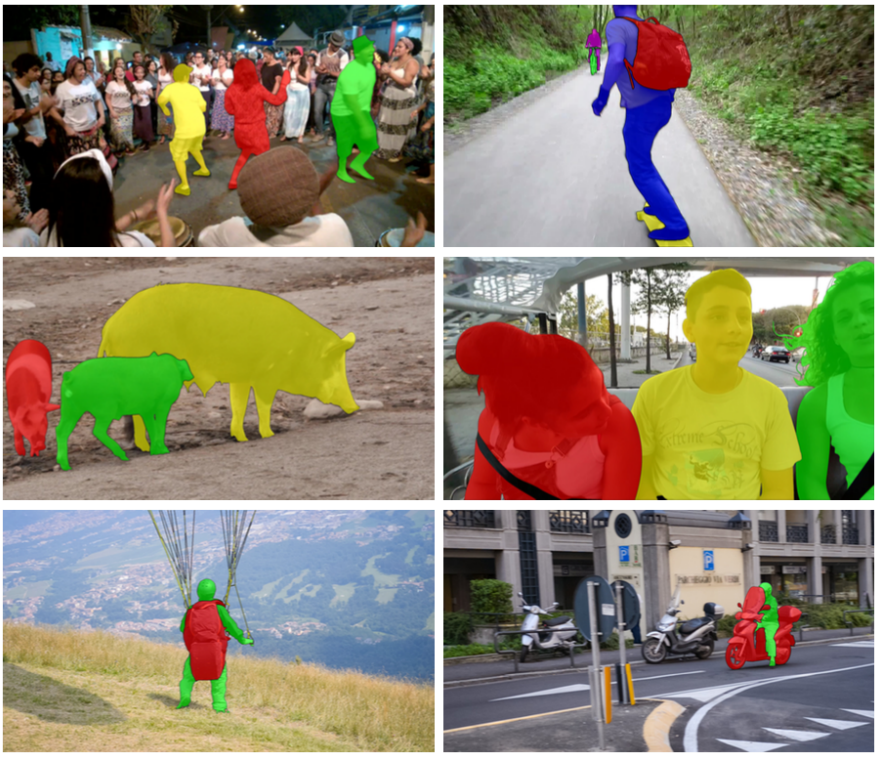


FIGURE 3.7: Example annotations of the DAVIS 2017 dataset. Notice that multiple types of objects are annotated (people, animals, backpacks, vehicles, etc.). Object classes are not provided.

3.2.1.2 Evaluation Metrics

In order to measure the quality of the predictions for the video human segmentation task, we also adopt the evaluation metric from the Davis 2017 Challenge. [Pont-Tuset et al., 2017]. We are using the intersection-over-union measure (also called the Jaccard Index) as a similarity measure between the estimated mask and the ground truth of a person in a particular frame:

$$IoU(P_h^f, G_h^f) = \frac{\left|P_h^f \cap G_h^f\right|}{\left|P_h^f \cup G_h^f\right|} \tag{3.2}$$

where G_h^f refers to the ground truth mask and P_h^f refers to the predicted mask for the human h in the frame f. $|M|$ refers to the number of pixels of mask M. A score of 1 denotes that the masks overlap perfectly (the masks are the same), while a score of 0 denotes that the masks have no common pixel.

FIGURE 3.8: Example annotations of the `DAVIS Persons` dataset.

On a set of sequences S, the performance of the metric is given by the mean intersection-over-union:

$$mIoU(S) = \frac{1}{|H_S|} \sum_{h \in H_S} \frac{1}{|F_h|} \sum_{f \in F_h} IoU(P_h^f, G_h^f) \tag{3.3}$$

where F_h is the set of all frames of the sequence which contains person h and H_S is the set of all humans contained in the entire set of sequences S. As in the evaluation procedure from [Pont-Tuset et al., 2017], F_h does not contain the first frame of the sequence from which we use supervision. Note that in order to calculate the $mIoU$ score, the IoU is averaged first across all frames of the sequence and then across all persons in the dataset. This implies that, on one hand, people in short sequences are as important as people in long sequences and, on the other hand, small-scale people are as important as large-scale people.

3.2.1.3 Training Details

For training, we use the same learning parameters as in [Khoreva et al., 2017a]. Optimization is done using mini-batch stochastic gradient descent with 10 images per batch. The learning rate policy is fixed with an initial learning rate of 10^{-3}, the momentum is $9 * 10^{-1}$ and the weight decay is set to $5 * 10^{-4}$.

Method	mIoU
FCIS Oracle	55.2
LucidTracker Baseline	56.6
`HumanMaskTracker`	**67.3**

TABLE 3.1: Comparison of our best tracker to the oracle and baseline methods on the `Davis Persons` dataset.

In terms of number of iterations, in the per-dataset training stage, we train for 40k iterations in all variants of the architecture. We have not noticed any significant increase in performance when training for more iterations. In the per-person training stage, where we fine-tune the model for a specific person, we train for 2k iterations.

As in [Khoreva et al., 2017b], we initialize the first convolutional weights corresponding to the extra input channels (previous mask, FCIS channel) using Gaussian noise.

In terms of computation time, training each per-person model takes around 3.5h (including data synthesis, computing the instance segmentation predictions over each synthetic RGB image, per-dataset training and per-person training). The per-dataset training is amortized over all persons in the dataset. At test time, the system runs at around 3.25s per frame (including the flow estimation with FlowNet2.0 [Ilg et al., 2017] ($\sim$ 0.5s) and the instance segmentation with FCIS [Yi Li and Wei, 2017] ($\sim$ 0.25s).

3.2.2 Key Results

Since to the best of our knowledge there is no prior work in tracking masks of people, we compare our main result against the state of the art method for video object segmentation (class agnostic) [Khoreva et al., 2017a] at the time of our experiments (June 2017), whose results we reproduce. We also create an oracle experiment to simulate a two-stage tracking by detection approach. Table 3.1 displays our main result and compares it to the baseline and the oracle, while Figure 3.10 shows qualitative results for a few persons.

3.2.2.1 Oracle Experiment

Our **FCIS oracle** experiment simulates a tracking by detection approach. It computes instance semantic segmentation in each frame of the video using FCIS. Then, for each person in the ground truth, it selects as prediction the FCIS human mask proposal which overlaps the most with the ground truth mask. If there is no such proposal or the overlap is 0, the prediction is the zero matrix.

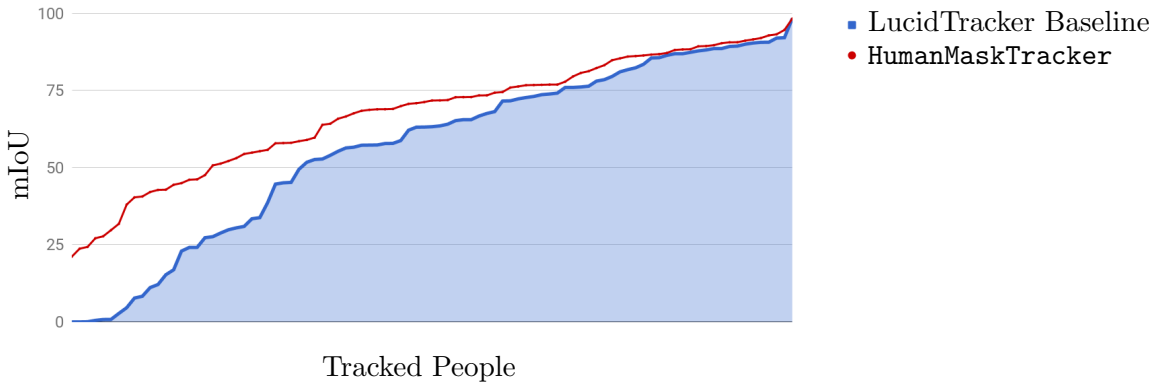


FIGURE 3.9: Comparison of the performance of `HumanMaskTracker` versus the LucidTracker Baseline. The tracked people are ordered by their performance on each method.

Having a perfect method of linking human masks over time (using ground truth), this oracle experiment shows us that current methods of human masks detectors are far from optimal.

3.2.2.2 Baseline

We propose a baseline method based on the LucidTracker [Khoreva et al., 2017a]. The method was designed for the task of video object segmentation on the DAVIS 2016 dataset [Perazzi et al., 2016] and is handling class agnostic tracking of single objects. We train the baseline to track the humans in `DAVIS Persons`.

The best performer in [Khoreva et al., 2017a] contains as input channels to the network not only I_t and $w(M^h_{t-1}, \mathcal{F}_t)$, but also $|\mathcal{F}_t|$ as the flow magnitude. On the `DAVIS Persons` dataset, using the flow magnitude is hurting tracking (see Section 3.2.3.3), so we do not use it as part of the baseline.

3.2.2.3 `HumanMaskTracker`

Our proposed system `HumanMaskTracker`, previously explained in Section 3.1, is related to the baseline method, but differs from it in two ways. First, it synthesizes data jointly for all people in a given sequence, being able to model occlusions (see Section 3.1.2.2). Second, it leverages semantic instance segmentation information about the human class by adding one more input channel $\mathcal{S}^h_t$ to the network (see Section 3.1.1).

FIGURE 3.10: Comparison of the predictions of the 3 methods on 3 different video sequences. We show predictions by sampling along the video duration.

Method	mIoU	Δ mIoU
LucidTracker Baseline	56.6	
LucidTracker Baseline + **New Synthesis Process**	62.5	≈ +6%

TABLE 3.2: Effect of the new synthesis process.

3.2.2.4 Comparison

By inspecting the qualitative results (see Fig. 3.10), we see that the masks predicted by the oracle are very coarse. This is mainly caused by the fact that FCIS was trained on the COCO dataset [Lin et al., 2014], whose annotations are not very fine. The annotations in the DAVIS dataset are, however, of very high quality. Although trained on synthetic data, both our method and the baseline show very fine pixel masks, hinting to the importance of leveraging high quality annotations. In addition to this, FCIS usually fails quite a lot in detecting all people in crowded scenes, which can also be observed in the boxing video shown in Figure 3.10.

Figure 3.9 shows the performance of the baseline and the `HumanMaskTracker` on all persons. Note that the proposed method mostly improves the performance in the tail of the plot, where tracking was underperforming. The predictions of both tracking approaches depend on the predictions in the previous frames. If tracking fails in the first frames, all consequent frames will most probably have poor predictions as well.

The second video in Figure 3.10 (lady with 2 dogs) shows the limitation of the baseline in the case of occlusion, when the mask wrongly spreads to the two dogs. The third video shows that the baseline is losing the mask of the person right from the beginning, mostly caused by the very small visible mask in the ground truth. The `HumanMaskTracker` performs well in both scenarios.

3.2.3 Analysis

In this section, we investigate the effect of different components in our tracker.

3.2.3.1 Effect of the New Synthesis Process

Table 3.2 shows the effect of replacing the **Lucid Data Dreaming** procedure used in the baseline with the proposed **Lucid Data Dreaming for Multiple People** synthesis process (Section 3.1.2.2). One can see the improvement is significant when modeling occlusions and inserting prior knowledge about locations of humans.

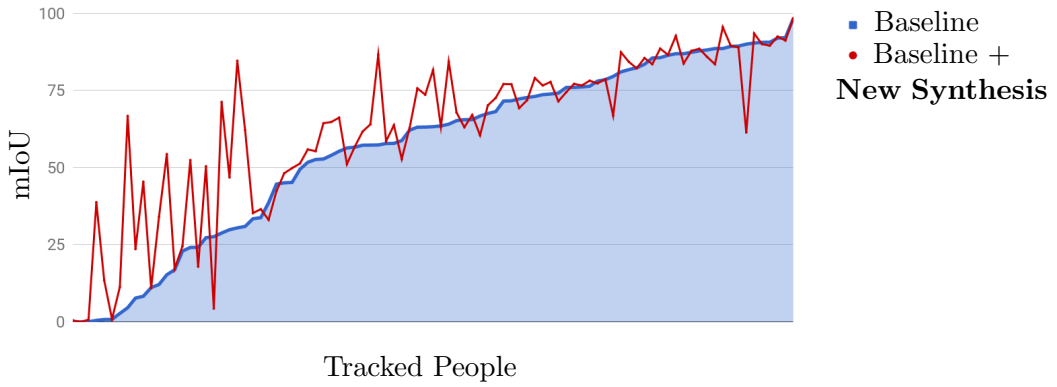


FIGURE 3.11: Performance difference when using **Lucid Data Dreaming for Multiple People**. The tracked people are ordered by their performance on the baseline.

FIGURE 3.12: Qualitative comparison between the baseline and the baseline trained with the new data synthesis method.

By analyzing Figure 3.11, we can see that the highest improvements happen for the tracks that were before failing on almost the entire sequence. When occlusions occur, masks can drift to other objects or can disappear. Either way, when the new synthesis process is used, the tracker becomes more robust to occlusions and this positively impacts the predictions in the frames following the occlusion.

Method	mIoU	Δ mIoU
`HumanMaskTracker` w/o $\mathcal{S}_t^h$	62.5	
`HumanMaskTracker`	**67.3**	$\approx +5\%$

TABLE 3.3: Effect of the $\mathcal{S}_t^h$ channel.

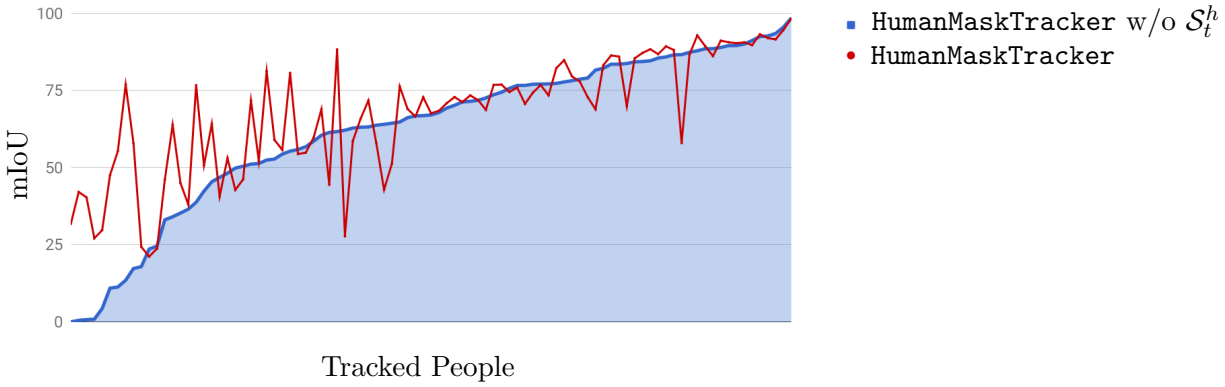


FIGURE 3.13: Performance difference when adding the $\mathcal{S}_t^h$ channel. The tracked people are ordered by their performance on the version without the $\mathcal{S}_t^h$ channel. The tracker "`HumanMaskTracker` w/o $\mathcal{S}_t^h$" is the same tracker as "Baseline + **New Synthesis**" from Section 3.2.3.1

When visually inspecting the effect of the new synthesis process (see Figure 3.12), we observe that the tracker is more robust to occlusions. Although we only simulated people to people occlusions or truncation by the image border, the tracker also becomes robust to occlusions by other objects (such as the tree in the biking sequence or the dogs in the second video).

3.2.3.2 Effect of $\mathcal{S}_t^h$

In Table 3.3 we show the additional 5% improvement that is brought by the extra $\mathcal{S}_t^h$ channel. The FCIS human proposals can be noisy, but they guide the tracker with general appearance information of the human class. Although our complex synthesis process generates many people in future hypothetical frames, the notion of human is limited to the appearance of the 93 people in the dataset from which we use supervision. The signal given by $\mathcal{S}_t^h$ can help in cases where the synthesis process did not capture enough appearance variability required in the future frames of the video.

In the top video showed in Figure 3.14, the tracker not using $\mathcal{S}_t^h$ has difficulties in segmenting only the human. Instead, the mask extends to the barrel in front of the blue cart. The given supervision mask (shown on the left) did not represent very well the person changing its size and showing different points of view across the video. When

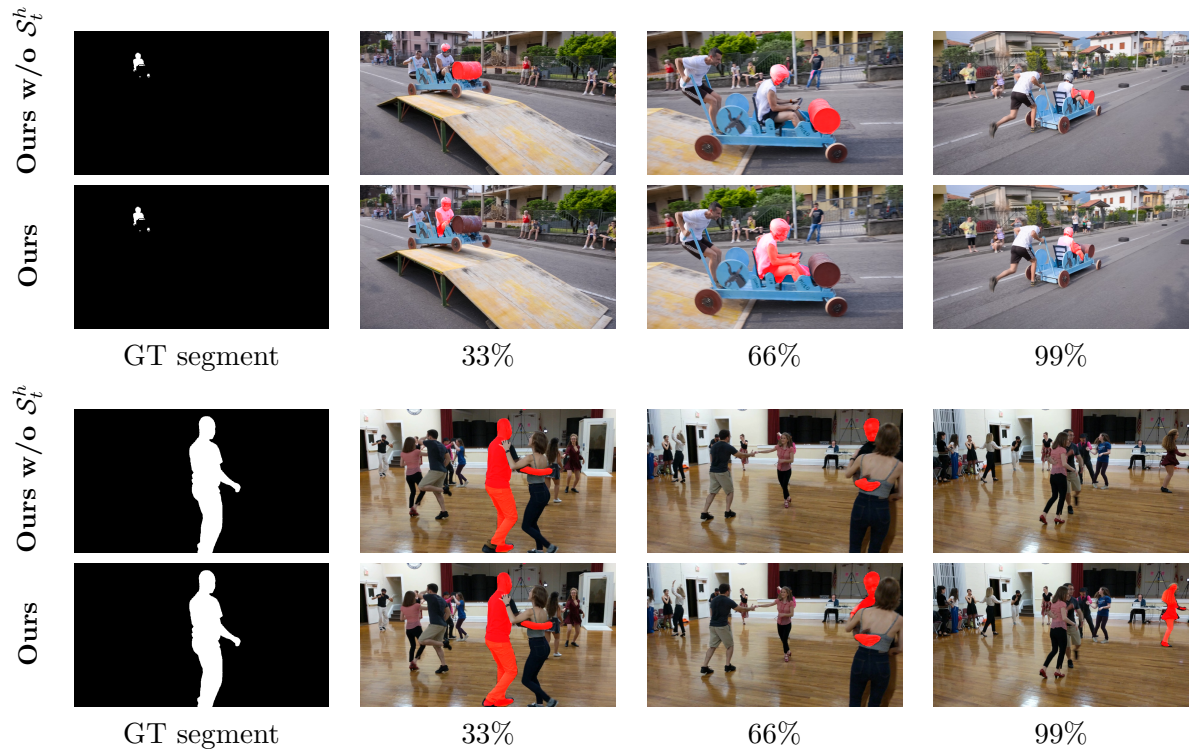


FIGURE 3.14: Qualitative comparison between the `HumanMaskTracker` w/o $\mathcal{S}_t^h$ and `HumanMaskTracker`

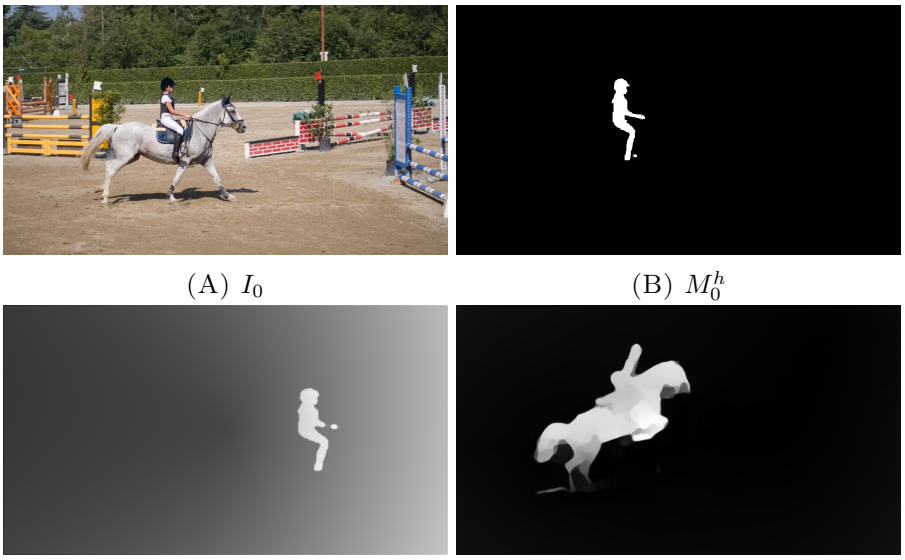
adding the additional $\mathcal{S}_t^h$ channel, the tracker receives enough signal about the human we are tracking and manages to correctly separate it from the background.

When analyzing Figure 3.13, it is worth noting that a few tracks are actually hurt by the new $\mathcal{S}_t^h$ channel. The bottom video showed in Figure 3.14 shows a case where the tracked person disappears from the scene and the mask is transferred to another person (the lady in the background who becomes visible while the man disappears). Because of the crowded scene, our algorithm for selecting $\mathcal{S}_t^h$ from the many FCIS human proposals is failing. One possible way of fixing this would be to teach the network to rely less on the $\mathcal{S}_t^h$ channel by adding some noise in that input during training. This would include random flips to other neighbouring FCIS proposals or zero masks. Whether or not this would be overall helpful has to be checked.

3.2.3.3 Using Flow Magnitude

As mentioned before, the baseline we select does not use the flow magnitude as an additional input channel.

In [Khoreva et al., 2017a], the synthesis process additionally computes the theoretical optical flow between consecutive synthetic frames. At test time, the optical flow is



(A) I_0 (B) M_0^h

(C) Example of simulated optical flow magnitude (D) Example of estimated optical flow magnitude

FIGURE 3.15: A sequence where multiple objects are moving together.

Variant	Per-person fine tuning	Warp with optical flow	mIoU
no w, no FT	✗	✗	55.8
no w	✓	✗	64.8
`HumanMaskTracker`	✓	✓	**67.3**

TABLE 3.4: Effect of flow warping and fine tuning on `HumanMaskTracker`

estimated with FlowNet2. In DAVIS 2016, only one object is annotated per sequence and it is usually the most salient one. Because of this, the magnitude of the FlowNet2 estimation is roughly the same as the segmentation mask of that object. Adopting the flow magnitude $|\mathcal{F}_t|$ as an additional input channel actually improves video object segmentation results, as shown in [Khoreva et al., 2017a].

On the other hand, in `DAVIS persons` many objects can be salient in each sequence (e.g. Fig. 3.15B). As our supervision comes only from masks of humans (e.g. Fig. 3.15B), the synthesis process can only simulate movements of humans (not other objects). In Fig. 3.15C, one can see an example of the simulated optical flow magnitude used for training. During test time, the optical flow magnitude is estimated with FlowNet2 and fires on all moving objects, not only humans (e.g. Fig 3.15D). This discrepancy between training and testing makes adopting $|\mathcal{F}_t|$ as an additional channel hurt the tracking results, so we do not include the channel in out method or the baseline.

3.2.3.4 Effect of Flow Warping and Fine Tuning

Table 3.4 shows, on one hand, the influence of warping the previous frame mask with the optical flow. At training time, we keep the same procedure, while at test time we use M^h_{t-1} instead of $w(M^h_{t-1}, \mathcal{F}_t)$ as an estimate of M^h_t. The 2.5% mIoU improvement when warping confirms that the optical flow contributes to a better estimation of M^h_t.

Second, the table shows a 9% decrease in mIoU when removing the last training stage, the per-person fine tuning. In this case, we use the same CNN model for all tracked persons in the dataset. When analyzing the qualitative results, we see that without fine tuning the tracker experiences drift to other people and objects or the mask disappears. This happens as the CNN has no knowledge of the appearance of the tracked person other than the M^h_{t-1} input, which is only an estimate. Fine tuning per person ensures the appearance of the person is encoded in the weights of the CNN as well.

3.2.4 Conclusion

The results show that the task of human segmentation tracking poses some challenges on the `DAVIS Persons` dataset. It is not yet clear that a tracking-by-detection approach can be successful, as current mask detection systems offer very coarse human mask proposals and are often ambiguous in crowded scenes.

We show that some recent techniques proposed for the task of video object segmentation can be successfully adopted for the task of human segmentation tracking. These include the mask refinement approach, training with synthetic data, fine-tuning per object and leveraging optical flow.

In addition, leveraging instance semantic segmentation proposals shows to provide the tracker with useful information about the people class. Moreover, modeling occlusions inside the data synthesis process proves to make the tracker more robust to these challenges.

Chapter 4

Human Pose Tracking

4.1 Method

In this section, we detail our approach on the Human Pose Tracking task. We assume the tracker receives one pose supervision for each test person from the center of their track (note this is not necessarily the center of the sequence).

4.1.1 Architecture

First, the motivation of having supervision in the middle of the track (and not in the first frame of the track) is that more frames will benefit from being close to the supervision (both previous and next frames). Tracking backwards from the mid-track to the beginning of the video is the same as reversing the order of the left sub-video and starting from the first frame and tracking forward on the left sub-video. Our proposed method does not make any assumption about the temporal order of the video (whether the time is increasing or decreasing). Because of this, for the purpose of simplicity, we will refer to the tracking scenario as starting from the first frame and proceeding frame by frame in forward direction.

Having available supervision in the form of one pose of each tracked person, the task of multi-person pose tracking can be approached as multiple separate single-person pose tracking problems.

To track the pose of a person, we propose using a convnet-based approach. For each new frame, we need to estimate the pose of a single target person, so we build upon the convnet architecture of an image-level single-person pose estimator. However, the frame can contain many people, so we need to direct the prediction towards the target

instance. To this end, inspired by advances in video object segmentation [Khoreva et al., 2017b], we use guidance from the appearance of the object (the RGB image), temporal consistency (the previous frame pose) and motion information (the optical flow) to predict the pose of the person of interest.

Formally, we predict the pose P_t^h of a human h in frame t as:

$$P_t^h = f(I_t, w(P_{t-1}^h, \mathcal{F}_t)) \tag{4.1}$$

where:

- f is the `HumanPoseTracker`, the function that we want to learn and use for frame by frame prediction. Depending on the training procedure, the function can be adapted *per-person* via fine-tuning (see Section 4.1.2.3).

  We model the function using a fully convolutional neural network designed for single-person pose estimation, DeeperCut [Insafutdinov et al., 2016], based on the ResNet-101 [He et al., 2016] architecture. The approach is a detection-based system, which generates a likelihood heatmap for each joint of the pose. At test time, the coordinate of each joint is determined by locating the point that has the maximum value in each likelihood heatmap (also called scoremap).

  Although pose estimation networks usually accept only RGB images as input, they can be easily extended to accept additional channels by increasing the depth of the filters of the first convolutional layer (from 3 to $3 + n$, where n is the number of extra channels).

- I_t is the **Current Image** at frame t, in RGB color mode.

- P_{t-1}^h is the **Previous Frame Pose** of person h predicted on image I_{t-1}. In the case where $t = 1$, the previous frame mask P_0^h is already available as ground truth due to the semi-supervised setting of the task.

  The motivation to condition the prediction of P_t^h by P_{t-1}^h comes from the temporal consistency assumption, i.e. objects do not change their pose configuration and position too much from one frame to another. Therefore, the previous predicted pose can be a good indicator of the location and pose configuration in frame t.

  We encode the previous frame pose P_{t-1}^h in a similar fashion as P_t^h, using one binary channel for each body part. Each channel contains a circular blob around the joint coordinate. If there is no coordinate for the particular joint, the respective channel will be the null matrix.

  Figure 4.1 shows two such poses. For visualization purposes, the joint channels are overlapped. Note that the blobs in P_t^h do not look perfectly circular, as the

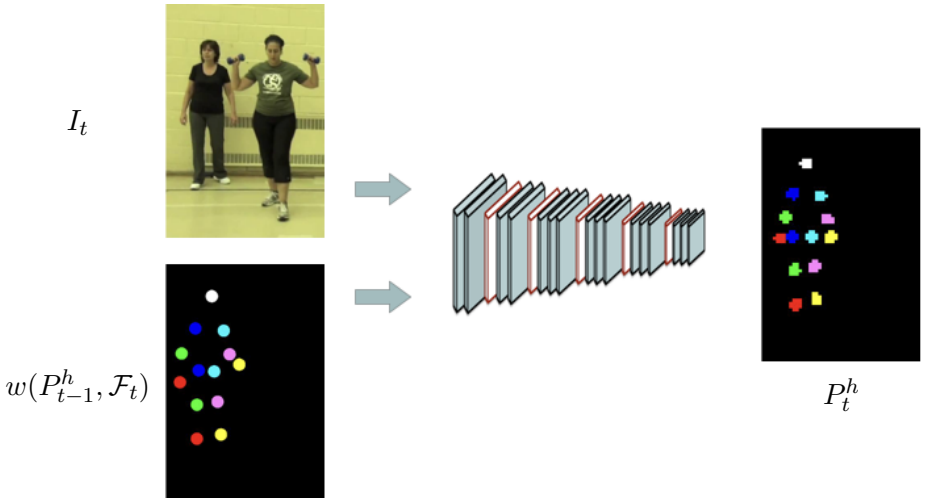


FIGURE 4.1: `HumanPoseTracker` architecture. Inputs include the RGB image I_t and the previous frame pose and motion information as $w(P^h_{t-1}, \mathcal{F}_t)$. The tracker is trained to predict P^h_t.

shape of the output is 8 times smaller than the input in each spatial dimension, due to the (8 px stride).

- $\mathcal{F}_t$ is the **Optical Flow** between image I_{t-1} and image I_t, which estimates the motion vector between the two frames. In our case, we compute it using FlowNet2.0 [Ilg et al., 2017], which is itself a trained convolutional neural network. It is estimated by only using the two RGB images I_{t-1} and I_t as input.

- w is an operation which **warps** the coordinates of each joint with the optical flow. We warp the previous frame pose P^h_{t-1} with the estimated optical flow $\mathcal{F}_t$ in order to have a better localized pose. While P^h_{t-1} is a good estimate of P^h_t, the warped version $w(P^h_{t-1}, \mathcal{F}_t)$ is even more indicative, as it incorporates motion information. This method of leveraging optical flow was previously shown to be effective in [Khoreva et al., 2017a] in the task of video object segmentation.

See Figure 4.1 for a visualization of the simplified `HumanPoseTracker` architecture.

Once trained, the `HumanPoseTracker` is initialized with the first frame ground-truth pose of the person it is supposed to track and then proceeds predicting the future poses frame by frame, following the temporal order. Note that the tracker is not fine-tuned from one frame to another, only its input changes.

4.1.2 Training Stage

The fact that the tracker architecture does not exploit longer temporal information (over many frames) comes to the benefit of the training procedure, which does not require

expensive densely annotated frames. Inspired by the approach of [Khoreva et al., 2017b], instead of using ground-truth poses P^h_{t-1}, P^h_t of consecutive frames, we only use ground truth annotations for P^h_t and instead synthesize $w(P^h_{t-1}, \mathcal{F}_t)$. This way, the network can be trained with static images only, using image-level pose estimation datasets. This is highly beneficial since video pose datasets contain limited appearance variability, while image-level pose datasets are usually large scale, covering many human appearances.

4.1.2.1 Previous Frame Pose Synthesis

The warped previous frame pose $w(P^h_{t-1}, \mathcal{F}_t)$ is, in essence, a noisy version of P^h_t, with noise introduced by the prediction in the previous frame and by the estimation of the optical flow. As in [Khoreva et al., 2017a], instead of synthesizing P^h_{t-1} and $\mathcal{F}_t$ separately, we synthesize $w(M^h_{t-1}, \mathcal{F}_t)$ directly.

To simulate the noise that we expect at test time, we shift each joint coordinate in P^h_t with a displacement vector. We do not model any dependence between joints, we assume there is no correlation between the noise of different body parts. The angle of the displacement vector is sampled uniformly, and its length is sampled from an exponential distribution. The sampling parameters were chosen based on statistics in the training set of the PoseTrack [Andriluka et al., 2017] dataset.

4.1.2.2 Data Processing and Augmentation

Each training sample corresponding to a human h contains an image I_t, the expected output P^h_t and the randomly generated $w(P^h_{t-1}, \mathcal{F}_t)$.

As the datasets that we are using for training and testing provide a head bounding box of the annotated person, we assume we also have access to it. The head bounding box is only used for estimating the height of the person. We do not consider the access to a head bounding box a requirement for our method, as one could also estimate the height of the person from its pose. As DeeperCut functions optimally when the person of interest has height of 340 px, we use the estimated height of the person to **rescale** the training sample to match this reference height. At test time, for each tracked human h, we rescale the entire sequence to the reference scale based on the height of h estimated from its one frame supervision. This assumes that the height of a person does not change drastically across the video sequence.

To help the tracker localize the person of interest, we adopt **cropping** around the target person. After padding the bounding box of the person's pose P^h_t with 250 px in each direction, we cut-out the outer parts of the image. At test time, we crop around

$w(P^h_{t-1}, \mathcal{F}_t)$, as P^h_t is not available. A tighter padding would impair the network from learning to track in crowded scenes, where more context is required to distinguish the target person.

The training procedure of DeeperCut contains a data augmentation step, which we also adopt for generating more training data. We apply random **rescaling** $\pm 15\%$, random **rotations** $\pm 30^\circ$ and random **flipping** around the vertical axis.

4.1.2.3 Training Modalities

The training procedure includes the following stages, in this order:

1. **ImageNet pre-training** - a common training stage in most CNNs; pre-training on a large classification dataset like ImageNet [Deng et al., 2009] ensures that the network is properly learning generic feature extractors that it might otherwise not be able to learn only with a smaller dataset;
2. **Offline training** - optimizes the CNN weights for the task of single person pose propagation; we train on an image level multi-person pose estimation dataset, where each human in each image becomes a training sample; the aforementioned data processing and augmentation is applied to the dataset;
3. **Per-person tuning** - trains the network to track a particular human h, using as training data only the supervision available for that specific person; the aforementioned data processing and augmentation is applied to the supervision; the tuning creates one tracker f_h per test person h, which is able to recognize the appearance of the person it was trained with; the idea is inspired from [Khoreva et al., 2017b] where it is used for the task of video object segmentation;

4.1.3 Testing Stage

After initializing the tracker of a person, we perform tracking twice, once in the forward direction, once in the backward direction. The output of the tracker is a full pose (one coordinate for each joint type) and a detection score for each joint. However, some of the joints may be incorrect. As we feed this pose back into the tracker, incorrect joints can propagate the error to the consequent frames. To drop incorrect joints, we threshold the joint detection scores. If the score of a keypoint is smaller than 0.7, we remove the keypoint from the predicted pose.

Ideally, the keypoint thresholding technique should remove all keypoints of the pose in the case when the tracked person completely disappears from the scene. However,

this condition is not always enough for stopping the tracking, so we propose two other stopping criteria:

- if all keypoint detection scores of a pose are smaller than 0.98, stop tracking; this strategy can be seen as a direct relaxation of the previous stopping criterium (stopping when all scores are smaller than 0.7);
- if the scene changes, stop tracking; to detect if a scene changes, we implement a simple scene change detector, which we test visually on the PoseTrack training set; the system checks consecutive frames, and it detects scene changes if the histogram and entropy of the two images vary significantly;

Note that the chosen detection score thresholds were cross-validated for maximizing the mMOTA score, with the condition of not hurting the mAP score.

4.2 Experiments

4.2.1 Experimental Setup

4.2.1.1 Dataset

To **test** our tracker, we require a dataset containing video-labeled multi-person poses. The only such dataset which contains relatively long sequences (more than 50 frames) is the recent PoseTrack [Andriluka et al., 2017], which we adopt for testing.

The sequences are selected as videos around certain single images in the MPII Human Pose dataset [Andriluka et al., 2014]. They include diverse human activities (recreational, occupational, household-related). The sequences are selected to contain extensive pose and appearance variation (see Figure 4.2), as well as a high amount of body motion. People can be highly occluded by other people or objects, and truncated. It can happen that people disappear and reappear in the scene. In addition to this, each video can contain multiple viewpoints (of the same scene or a different one) and in this case the track ID of the person changes when the shot is switched.

As our method requires one pose supervision for each track, we report our findings on the validation set, whose annotations are publicly available. It contains 50 videos with a total of $\approx$ 700 annotated people. The length of the sequences ranges from 65 to 298 frames. In each video, 30 frames are densely annotated and all the other frames (to the left and to the right of this interval) contain annotations only every 4 frames. This amounts to a total of $\approx$ 19k poses.

FIGURE 4.2: Example annotations from the PoseTrack dataset [Andriluka et al., 2017].

For each track ID in the test set, we choose as supervision the pose in the middle of the track. In case the middle of the track is a frame which is not annotated, or is annotated but the person is not in the frame, we select as the middle of the track the closest frame which contains the particular track ID. We do not use this mid-track point as anything other than a starting point for tracking in the two directions. For example, our method does not use the fact that it is located in the middle of the track and implicitly the tracking in the two directions will run for roughly the same number of frames.

The pose in PoseTrack contains 15 body joint types: head, nose, neck, shoulder, elbows, wrists, hips, knees and ankles.

To **train** the tracker, our method requires only image-level multi-person poses. There are basically two large scale datasets available: MPII Human Pose [Andriluka et al., 2014] and MS COCO Keypoints [Lin et al., 2014]. Although COCO contains the largest

number of people, we train on MPII-Pose for its images come from the same domain as the PoseTrack data that we use for testing.

Alternative approaches, that we did not experiment with, would have been to train on COCO and fine-tune on MPII-Pose, or train on COCO and fine-tune on MPII-Pose and the PoseTrack training set. On their own, PoseTrack Train does not contain enough appearance variability and COCO is from a different domain to the testing set.

The MPII Human Pose training set that we use for training contains $\approx$ 29k poses. The pose is represented by 14 body joint types, which are the same as in PoseTrack, with the exception of the nose, which is not represented in MPII Human Pose. We remove the nose joint from the PoseTrack annotations and report all results on the 14 joint types common in both datasets.

For **collecting the statistics** required to sample noise in the previous frame pose synthesis, we use the training set of PoseTrack, as it is densely annotated.

4.2.1.2 Evaluation Metrics

We adopt the evaluation metrics proposed in the PoseTrack benchmark [Andriluka et al., 2017]: Average Precision (AP) for multi-person pose estimation and Multiple Object Tracker Accuracy (MOTA) for multi-person pose tracking. As our method approaches the multi-person task separately for each person, we find it relevant to also compute a single-person pose estimation metric, namely the head-normalized probability of correct keypoint (PCKh).

Computing these metrics requires that a head bounding box annotation is available for each ground truth pose. It is used to compute a person-specific distance threshold, set to 30% of the length of the diagonal of the bounding box. A predicted keypoint is correctly localized if its distance to the ground truth (GT) location lies within this distance. The PCKh score between a GT pose and a predicted pose is calculated as the number of correctly localized joints divided by the total number of GT joints.

- **AP** is used to measure the per-frame performance of a multi-person pose estimator. It requires joint detection scores as input, but no track IDs of the poses. First, a greedy matching is computed between the predicted poses and the GT poses. Each predicted pose is assigned to the closest GT pose based on the highest PCKh. To ensure that only one predicted pose can be assigned to each GT pose, the predicted pose with the highest PCKh is selected as the match for the GT pose.

The remaining predicted poses are counted as false positives. With this matching, for each body part, the AP score is computed using the detection score given as input. Mean AP (mAP) is calculated as the AP score averaged over all body joint types.

- **MOTA** is used to evaluate multi-person pose tracking. It requires track IDs as input, but no joint detection scores. First, for each joint type in each frame, all distances between predicted joints and corresponding GT joints are computed. All (GT,prediction) pairs in which the prediction is correctly localized as GT (based on the PCKh threshold) are considered for global matching. Global matching, which takes into account the track IDs of the pairs, minimizes the total assignment distance. This produces a match between GT track IDs and predicted track IDs for each body joint. With this match, Multiple Object Tracker Accuracy (MOTA) is computed for each keypoint type, as well as their average, mMOTA.

- **PCKh** is used to evaluate per-frame single-person pose estimation. It requires neither joint detection scores, nor track IDs of the predicted poses. First, in each frame, matching is performed between the predicted poses and the GT poses as in the process of computing the mAP. With this matching, for each keypoint type, PCKh is computed as the probability of the GT joint type to be correctly predicted (within the PCKh threshold). mPCKh is computed as the average over all body joint types.

As discussed before, our method requires supervision at test time in the form of one pose per track. We do not remove these poses from the GT, we keep them in the evaluation. On the other hand, we remove the nose keypoint from the GT, since our tracker was trained without it, as described in Section 4.2.1.1.

4.2.1.3 Training Details

For training, we use the same learning parameters as in the TensorFlow [Abadi et al., 2016] implementation of [Insafutdinov et al., 2016]. Optimization is done using stochastic gradient descent with 1 image per batch.

In the offline training stage, we start with the learning rate $lr = 0.005$ for 10k iterations, then $lr = 0.02$ for 420k iterations, $lr = 0.002$ for 300k iterations and $lr = 0.001$ for 300k iterations, which amounts to a total of 1030k iterations. In the per-person tuning stage, where we train the model for a specific person, we train for 1k iterations using $lr = 0.002$. We experimented with a larger number of iterations and a smaller learning rate, but we did not observe any improvement.

Variant	mAP (Δ)	mMOTA (Δ)	mPCKh (Δ)
`HumanPoseTracker`	**72.6**	**64.6**	**76.3**
Pose Tracking winner [Girdhar et al., 2017]	59.7 (-12.9)	54 (-10.6)	65.6 (-10.7)
Pose Estimation winner [Zhu et al., 2017]	67.2 (-5.4)	20.1 (-44.5)	71.8 (-4.5)

TABLE 4.1: Comparison of our best tracker to the winners of the PoseTrack challenge [Andriluka et al., 2017]. Note that the challenge winners do not assume supervision at test time.

We initialize the first convolutional weights corresponding to the extra input channels $w(P^h_{t-1}, \mathcal{F}_t)$ with the already ImageNet trained weights corresponding to I^h_t.

In terms of computation time, offline training takes around 4 days, while fine-tuning lasts around 6 minutes per person. At test time, the system runs at around 0.7s per frame, including the optical flow estimation with FlowNet2.0 [Ilg et al., 2017]. Our experiments run on a single NVIDIA Tesla K40 GPU with 12 GB RAM.

4.2.2 Key Results

Since to the best of our knowledge there is no prior work addressing the semi-supervised scenario of multi-person pose tracking, we compare our main result against the unsupervised approaches that have recently achieved state of the art results on the PoseTrack dataset. Though not a fair comparison, it helps to understand how much a pose tracker can benefit from test-time supervision.

4.2.2.1 PoseTrack Challenge Winners

As all the other methods proposed in the PoseTrack challenge, the **pose tracking winner** [Girdhar et al., 2017] proposes a tracking-by-detection framework. First, multi-person pose estimation is performed on the image level (in this case using Mask-RCNN [He et al., 2017]) to detect all poses in each frame. Secondly, the poses are linked across the frames into tracks. In the case of the tracking winner [Girdhar et al., 2017], matching is performed using the Hungarian Algorithm, where the cost of matching is computed as the overlap between bounding boxes of the persons in different frames. Table 4.1 shows the performance of this method on both pose tracking and pose estimation tasks.

The **pose estimation winner** [Zhu et al., 2017] builds upon the Part Affinity Fields framework [Cao et al., 2016] and proposes several improvements. These include modifying the structure of the kinematic tree, using advances in semantic segmentation to improve the network architecture, as well as training on external training data. Its performance on the multi-person pose estimation task is shown in Table 4.1.

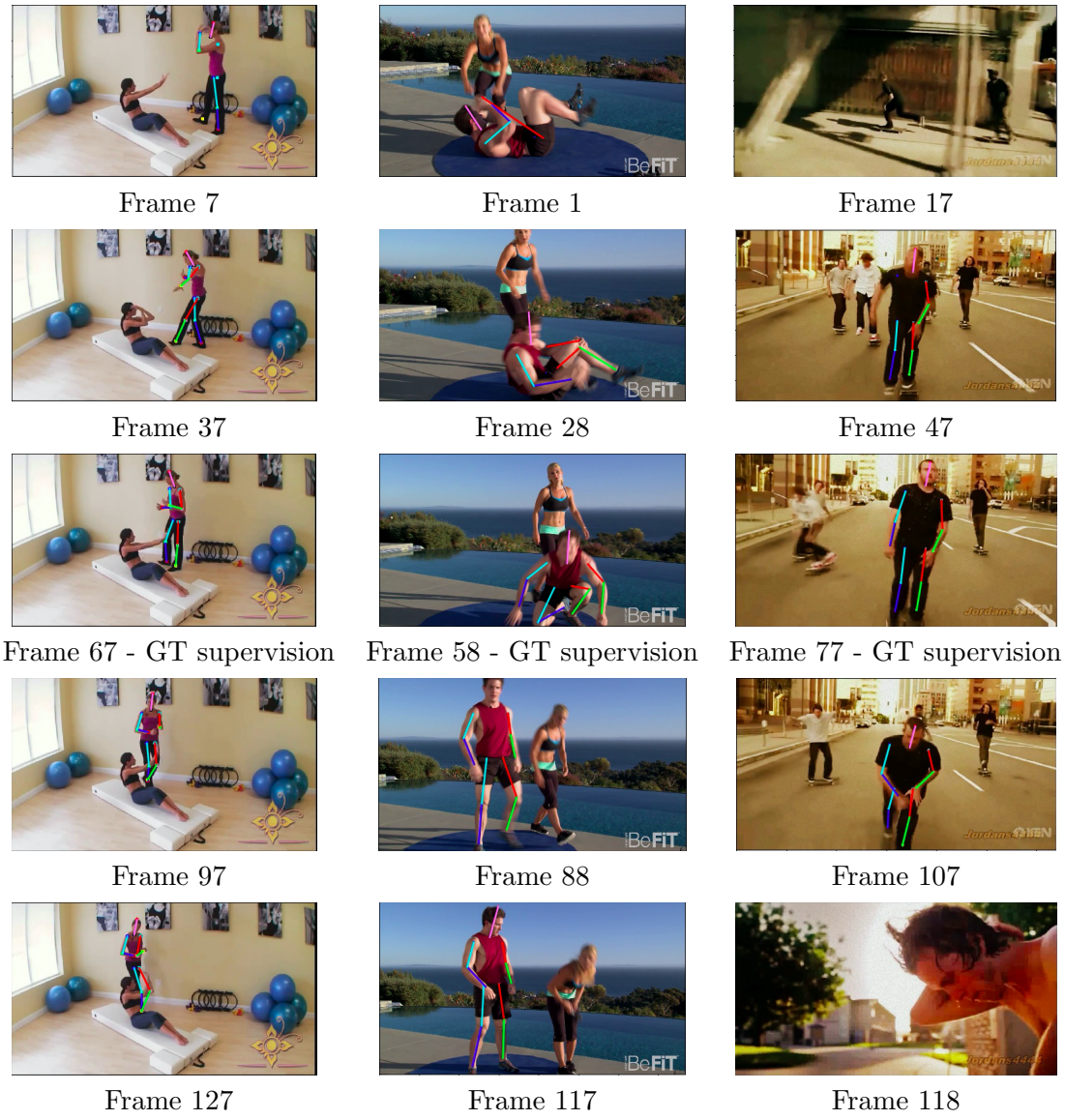


FIGURE 4.3: Qualitative results of `HumanPoseTrack`.

4.2.2.2 `HumanPoseTracker`

Our proposed system `HumanPoseTracker`, previously explained in Section 4.1, can perform pose tracking in one stage, but at the cost of requiring one pose supervision for initializing each track. Qualitative results are shown in Figure 4.3. Note how our tracker can handle high variations of the pose (left and mid sequence), fast movements (mid sequence), as well as scene changes (2 changes in the right sequence where the tracker manages to stop).

The tracker is, however, not perfect. Frame 127 in the left sequence also displays a failure case at the feet and knees joints, right after they were occluded by the person in front of them. In this case, the predictions wrongly move together with the person that

Variant	mAP (Δ)	mMOTA (Δ)	mPCKh (Δ)
`HumanPoseTracker`	**72.6**	**64.6**	**76.3**
NO Flow Warping	65.6 (-7.0)	58.4 (-6.2)	69.1 (-7.2)
NO Per-Person Fine Tuning	66.3 (-6.3)	56.0 (-8.6)	70.6 (-5.7)

TABLE 4.2: Ablative Study: effect of flow warping and per-person fine-tuning.

occluded them. This is caused by the flow, which moves the previous predicted joints in the direction of the movement of the person in front of them. A possible fix could be to introduce more noise in the synthesis of $w(P^h_{t-1}, \mathcal{F}_t)$ during training. Frame 1 in the mid sequence also shows a failure case in the estimation of the pose, which is not surprising given the highly uncommon articulated pose of the person.

4.2.2.3 Comparison

The quantitative comparison between our method and the PoseTrack challenge winners is shown in Table 4.1. By integrating one supervision per track during test time, our method manages to significantly improve both single-image multi-person pose estimation and multi-person pose tracking results. The highest mAP score is improved by 5.4, while the highest mMOTA increases by 10.6.

Successfully addressing both tasks at the same time is challenging, as it can be seen from the scores obtained by the winning methods on the challenges that they do not optimize for. The Pose Tracking winner is 7.5 points behind the state of the art on the Pose Estimation task, while the winner of the Pose Estimation challenge is 33.9 points behind the best mMOTA result.

The mPCKh score, a single-person pose estimation metric insensitive to track IDs or detection scores, also shows improvement for our method, indicating that localization of keypoints is more accurate.

4.2.3 Analysis

In this section, we present an extensive ablation study, which validates various design choices of our model.

4.2.3.1 Effect of Flow Warping and Fine Tuning

Table 4.2 shows, on one hand, the influence of warping the previous frame mask with the optical flow. At training time, we keep the same procedure, while at test time we

Variant	mAP (Δ)	mMOTA (Δ)	mPCKh (Δ)
`HumanPoseTracker`	**72.6**	**64.6**	76.3
NO Joint Thresholding (thresh=0.7)	72.6 (0)	41.3 (-23.3)	**79.4** (+3.1)
NO Stop Tracking when all joint scores ≤ 0.98	72.5 (-0.1)	62.9 (-1.7)	76.4 (+0.1)
NO Stop Tracking when scene changes	72.4 (-0.2)	62.2 (-2.4)	76.9 (+0.6)

TABLE 4.3: Ablative Study: effect of keypoint thresholding and stopping tracking criteria

use P^h_{t-1} instead of $w(P^h_{t-1}, \mathcal{F}_t)$ as an estimate of P^h_t. The 7.9 mAP, 6.1 mMOTA and 7.2 mPCKh improvements when warping confirm that the optical flow contributes to a better estimation of P^h_t.

To avoid warping with the flow, we would have to simulate P^h_{t-1} from P^h_t during training time, which would require modeling the dependencies between keypoints in the same image as well as across consecutive frames. For its simplicity and effectiveness, we only experimented with flow warping.

Second, the table shows similar decreases in performance (on all metrics) when removing the last training stage, the per-person fine tuning. In this case, we use the same CNN model for all tracked persons in the dataset. When analyzing the qualitative results, we see that without fine tuning the tracker experiences drift to other people and objects or the pose disappears. This happens as the CNN has no knowledge of the appearance of the tracked person other than the P^h_{t-1} input, which is only an estimate. Fine tuning per person ensures the appearance of the person is encoded in the weights of the CNN as well.

4.2.3.2 Joint Thresholding and Stopping Criteria Influence

Table 4.3 shows, on one hand, the influence of removing keypoints by thresholding their detection scores. During cross-validating the threshold, we observed that by increasing the threshold within the $[0, 0.95]$ interval, the mAP score is decreasing because of an increase in false negatives, while the mMOTA increases due to fewer incorrect detections. For this reason, the 0.7 threshold was selected as it maximizes the mMOTA score while still not degrading the mAP score.

We can see that thresholding joint detections has a very high positive impact on the tracking results (increase by 23.3 on mMOTA). Thresholding does not only remove incorrect detections in the frame where it is performed, but in our case it also prevents the tracker from propagating the errors to the next frames.

The two stopping criteria we adopt are also improving the tracking results by removing false positive people detections. By visually inspecting their effect, we saw that they can also introduce false negatives. This can happen when the scene change detector wrongly classifies a frame as a scene transition point. Note that the scene change detector we use is very simple and it does not involve any learning, so there is room for improvement in this stage of the pipeline. Also, our method is not designed to restart tracking once a person who disappeared from the scene returns, so false negatives can also be introduced in this case.

Note that the mPCKh score is actually hurt by the thresholding and the stopping criteria. This is expected, as mPCKh does not punish false positives detections. Therefore, removing false positives does not influence the mPCKh score, while removing true positives decreases it.

4.2.4 Leveraging Additional Supervision

Having shown significant improvement in both pose tracking and pose estimation using only one pose supervision per track, we now investigate the impact of using more supervision during testing.

We design our experiments to use more and more supervision. For each track ID, we collect one pose supervision in the middle of the track as described in the original setting. Afterwards, we proceed forward and backward starting with the middle of the track and collect supervision for the specific track ID every k frames (if available). The validation set of PoseTrack is densely annotated for only 30 frames in the middle of each sequence, while the tails are coarsely annotated, every 4 frames. To make sure the supervisions we collect are distanced equally from each other, we choose k as a power of 4. A large sampling rate ($k = 1000$) for the sequences in PoseTrack implies that only the middle of the track is used as supervision, which is our original setting. We compute the supervision percentage as the percentage of GT annotations used as supervision, averaged over all track IDs. In the case of PoseTrack, the supervision percentage is not $1/k$, as the videos are not densely annotated. The supervision percentage is relevant as we do not eliminate the supervision poses from the evaluation.

For each track ID, all the collected supervision is used to fine-tune the offline-trained tracker, which creates one model per tracked person. The alternative of training one model for each pose supervision would be too computationally expensive.

At run-time, for each track ID, for each pose supervision, we initialize the tracker and proceed tracking backwards and forward for $k/2$ frames with the same thresholding

Sampling Rate k	Supervision Percentage	mAP	mMOTA	mPCKh
4	49.68	97	95.5	97.2
8	24.86	92.4	88.6	93.1
16	12.42	86.7	81.2	88.3
32	6.48	79.2	72.3	82
64	3.58	72.6	65.3	76.3
1000	3.37	72.6	64.6	76.3

TABLE 4.4: The effect of using more supervision during tracking.

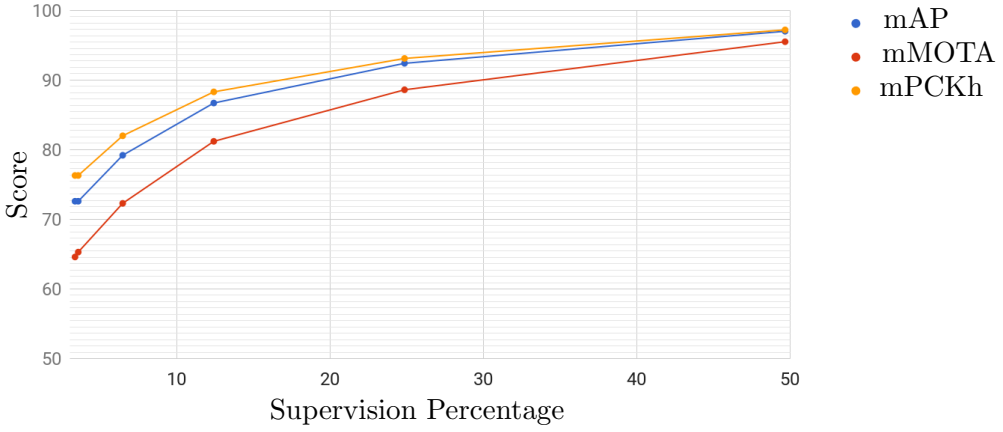


FIGURE 4.4: The effect of using more supervision during tracking. The x axis represents the percentage of GT used as supervision during testing, averaged across all tracks. The y axis represents the score of the particular plotted metric.

and stopping criteria techniques as described before. To obtain the final track, we merge all tracks generated by the supervision poses of a particular person.

The results of the experiments are shown in Table 4.4, and plotted in Figure 4.4. We can see that the performance of the system increases with more supervision, from 72.6 mAP, 64.6 mMOTA and 76.3 mPCKh when using only one pose per track, up to 97 mAP, 95.5 mMOTA and 97.2 mPCKh when using supervision every 4 frames. These results show that for a dataset annotated coarsely, every 4 frames, our system could be used to predict very accurate poses in between the annotated frames, turning it into a densely annotated dataset.

4.2.5 Conclusion

Our experiments show that the task of multi-person pose tracking can benefit significantly from using one pose supervision per track during test time. The method manages to notably improve both pose estimation and pose tracking results at the same time.

Further parameter tuning can make these improvements even higher when targeting one task at a time.

We show the effectiveness of each of our design choices through an extensive ablation study. Fine-tuning per object and leveraging optical flow, techniques proposed for the task of video object segmentation, prove to be highly effective for supervised pose tracking as well. Also, removing unconfident joint detections is shown to be important for pose tracking in general and for our propagation method in particular.

Finally, we show that our method can be extended to leverage more supervision during test time and it has the potential to be used for generating dense from sparse annotations in videos.

Chapter 5

Conclusion

To summarize, our work addresses the multi-person tracking task with two types of representation: body pose and segmentation mask. We explore these scenarios in the semi-supervised setting, where one available annotation is available per person during test time. More complex representations of people (segmentation mask and body pose) can provide richer understanding of visual scenes, and methods that leverage supervision during test time should be developed for the cases when supervision is available.

We propose `HumanMaskTracker` for the task of video human segmentation. Our approach builds on recent techniques proposed for the task of video object segmentation. These include the mask refinement approach, training with synthetic data, fine-tuning per object and leveraging optical flow. In addition, we propose leveraging instance semantic segmentation proposals to give the tracker a better notion about the human class. Moreover, we propose modeling occlusions inside the data synthesis process. This proves to make the tracker more robust to the challenges of occlusion and disocclusion.

For the task of semi-supervised multi-person pose tracking, we propose the method `HumanPoseTracker`. We show that the task of multi-person pose tracking can benefit significantly from using one pose supervision per track during test time. The method actually manages to notably improve both pose estimation and pose tracking results at the same time. We show the effectiveness of each of our design choices. Fine-tuning per object and leveraging optical flow, techniques proposed for the task of video object segmentation, prove to be highly effective for supervised pose tracking as well. Also, removing unconfident joint detections is shown to be important for pose tracking in general and for our propagation method in particular. A promising application of our work is shown by extending our method to leverage more supervision during test time.

Bibliography

[Abadi et al., 2016] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

[Andriluka et al., 2017] Andriluka, M., Iqbal, U., Milan, A., Insafutdinov, E., Pishchulin, L., Gall, J., and Schiele, B. (2017). Posetrack: A benchmark for human pose estimation and tracking. *arXiv preprint arXiv:1710.10000*.

[Andriluka et al., 2014] Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693.

[Andriluka et al., 2008] Andriluka, M., Roth, S., and Schiele, B. (2008). People-tracking-by-detection and people-detection-by-tracking. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

[Bookstein, 1989] Bookstein, F. L. (1989). Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence*, 11(6):567–585.

[Cao et al., 2016] Cao, Z., Simon, T., Wei, S.-E., and Sheikh, Y. (2016). Realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1611.08050*.

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

[Criminisi et al., 2004] Criminisi, A., Pérez, P., and Toyama, K. (2004). Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 13(9):1200–1212.

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

[Girdhar et al., 2017] Girdhar, R., Gkioxari, G., Torresani, L., Ramanan, D., Paluri, M., and Tran, D. (2017). Simple, efficient and effective keypoint tracking.

[He et al., 2017] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. *arXiv preprint arXiv:1703.06870*.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[Held et al., 2016] Held, D., Thrun, S., and Savarese, S. (2016). Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, pages 749–765. Springer.

[Ilg et al., 2017] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Insafutdinov et al., 2017] Insafutdinov, E., Andriluka, M., Pishchulin, L., Tang, S., Levinkov, E., Andres, B., Schiele, B., and Campus, S. I. (2017). Arttrack: Articulated multi-person tracking in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 4327.

[Insafutdinov et al., 2016] Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., and Schiele, B. (2016). Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer.

[Iqbal et al., 2017] Iqbal, U., Milan, A., and Gall, J. (2017). Posetrack: Joint multi-person pose estimation and tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Khoreva et al., 2017a] Khoreva, A., Benenson, R., Ilg, E., Brox, T., and Schiele, B. (2017a). Lucid data dreaming for multiple object tracking. In *arXiv preprint arXiv: 1703.09554*.

[Khoreva et al., 2017b] Khoreva, A., Perazzi, F., Benenson, R., Schiele, B., and Sorkine-Hornung, A. (2017b). Learning video object segmentation from static images. In *30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, Honolulu, HI, USA.

[Krähenbühl and Koltun, 2011] Krähenbühl, P. and Koltun, V. (2011). Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117.

[Lin et al., 2014] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

[Long et al., 2015] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440.

[Milan et al., 2014] Milan, A., Roth, S., and Schindler, K. (2014). Continuous energy minimization for multitarget tracking. *IEEE transactions on pattern analysis and machine intelligence*, 36(1):58–72.

[Nam et al., 2016] Nam, H., Baek, M., and Han, B. (2016). Modeling and propagating cnns in a tree structure for visual tracking. *arXiv preprint arXiv:1608.07242*.

[Noh et al., 2015] Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528.

[Papandreou et al., 2017] Papandreou, G., Zhu, T., Kanazawa, N., Toshev, A., Tompson, J., Bregler, C., and Murphy, K. (2017). Towards accurate multi-person pose estimation in the wild. *arXiv preprint arXiv:1701.01779*.

[Perazzi et al., 2016] Perazzi, F., Pont-Tuset, J., McWilliams, B., Van Gool, L., Gross, M., and Sorkine-Hornung, A. (2016). A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–732.

[Pont-Tuset et al., 2017] Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., and Van Gool, L. (2017). The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

[Tang et al., 2017] Tang, S., Andriluka, M., Andres, B., and Schiele, B. (2017). Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3539–3548.

[Yi Li and Wei, 2017] Yi Li, Haozhi Qi, J. D. X. J. and Wei, Y. (2017). Fully convolutional instance-aware semantic segmentation.

[Zhu et al., 2017] Zhu, X., Jiang, Y., and Luo, Z. (2017). Multi-person pose estimation for posetrack with enhanced part affinity fields.