HAL
open science

Learning Motion Patterns in Videos

Pavel Tokmakov, Karteek Alahari, Cordelia Schmid

To cite this version:

Pavel Tokmakov, Karteek Alahari, Cordelia Schmid. Learning Motion Patterns in Videos. CVPR 2017 - IEEE Conference on Computer Vision & Pattern Recognition, Jul 2017, Honolulu, United States. hal-01427480v1

HAL Id: hal-01427480

https://hal.science/hal-01427480v1

Submitted on 5 Jan 2017 (v1), last revised 11 Apr 2017 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Learning Motion Patterns in Videos

Pavel Tokmakov Karteek Alahari Cordelia Schmid

Inria*

Abstract

The problem of determining whether an object is in motion, irrespective of the camera motion, is far from being solved. We address this challenging task by learning motion patterns in videos. The core of our approach is a fully convolutional network, which is learnt entirely from synthetic video sequences, and their ground-truth optical flow and motion segmentation. This encoder-decoder style architecture first learns a coarse representation of the optical flow field features, and then refines it iteratively to produce motion labels at the original high-resolution. The output label of each pixel denotes whether it has undergone independent motion, i.e., irrespective of the camera motion. We demonstrate the benefits of this learning framework on the moving object segmentation task, where the goal is to segment all the objects in motion. To this end we integrate an objectness measure into the framework. Our approach outperforms the top method on the recently released DAVIS benchmark dataset, comprising real-world sequences, by 5.6%. We also evaluate on the Berkeley motion segmentation database, achieving state-of-the-art results.

1. Introduction

The task of analyzing motion patterns has a long history in computer vision [3, 8, 15, 29, 36, 38, 43]. This includes methods for motion estimation [3, 36], scene [38] and optical [8, 15] flow computation, video segmentation [29, 43]; all of which aim to capitalize on or estimate motion cues in scenes. Despite this progress, the fundamental problem of identifying if an object is indeed moving, irrespective of camera motion, remains challenging. In this paper, we make significant advances to address this challenge, with a novel CNN-based framework to automatically learn motion patterns in videos, and use it to segment moving objects; see sample results in Figure 1.

To illustrate the task, consider Figure 2, a sequence from the FlyingThings3D dataset [26]. It depicts a scene generated synthetically, involving a moving camera (can be easily observed by comparing the top left corners of the images (a) and (b)), with objects in motion, e.g., the three large objects in the centre of the frame (which are easier to spot in the ground-truth segmentation (d)). The goal of our work is to study such motion patterns in video sequences (using optical flow field (c)), and to learn to distinguish real motion of objects from camera motion. In other words, we target the moving object segmentation in (d).

Figure 1. Results on the DAVIS dataset. Left: Optical flow field input to our MP-Net, computed with [6]. Right: Our segmentation result overlaid on the video frame. Note that our approach accurately segments moving objects, and learns to distinguish between object and camera motions (seen in the flow fields).

The core of our approach is a trainable model, motion pattern network (MP-Net), for separating independent object and camera motion, which takes optical flow as input and outputs a per-pixel score for moving objects. Inspired by recent fully convolutional networks (FCNs) [8, 24, 33], we propose a related encoder-decoder style architecture to accomplish this two-label classification task. The network is trained from scratch with synthetic data [26]. Pixel-level ground-truth labels for training are generated automatically (see Figure 2(d)), and denote whether each pixel has moved in the scene. The input to the network is flow fields, such as the one shown in Figure 2(c). More details of the network, and how it is trained are provided in Section 3. With this training, our model learns to distinguish motion patterns of objects and background. We then demonstrate the efficacy of the learnt model on the moving object segmentation task (see §6). To tackle this task, we integrate an

*Thoth team, Inria, Laboratoire Jean Kuntzmann, Grenoble, France.

(a) (b) (c) (d)

Figure 2. (a,b) Two example frames from a sequence in the FlyingThings3D dataset [26]. The camera is in motion in this scene, along with four independently moving objects. (c) Ground-truth optical flow of (a), which illustrates motion of both foreground objects and background with respect to the next frame (b). (d) Ground-truth segmentation of moving objects in this scene.

objectness mesure into our approach, which otherwise also detects non-object motions such as moving water or smoke. Specifically, we evaluate on the densely annotated video segmentation (DAVIS) [30] and the Berkeley motion segmentation (BMS-26) [3, 5, 37] datasets, both comprising real-data sequences. We obtain state-of-the-art results on these challenging datasets. In particular, we outperform previous video-level methods by over 5.6%, on the intersection over union score, on DAVIS, despite operating only on the frame level.

2. Related Work

Our work is related to the following tasks dealing with motion cues: motion and scene flow estimation, and video object segmentation. We will review the most relevant work on these topics, in addition to a review of related CNN architectures in the remainder of this section.

Motion estimation. Early attempts for estimating motion have focused on geometry-based approaches, such as [36], where the potential set of motions is identified with RANSAC. Recent approaches have relied on other cues to estimate moving object regions. For example, Papzouglou and Ferrari [29] first extract motion boundaries by measuring changes in optical flow field, and use it to estimate moving regions. They also refine this initial estimate iteratively with appearance features. This approach produces interesting results, but is limited by its heuristic initialization. We show that incorporating our learning-based motion estimation into it improves the results significantly (see Table 4).

Narayana *et al*. [27] use optical flow orientations in a probabilistic model to assign pixels with labels that are consistent with their respective real-world motion. This approach assumes pure translational camera motion, and is prone to errors when the object and camera motions are consistent with each other. Bideau *et al*. [3] presented an alternative to this, where initial estimates of foreground and background motion models are updated over time, with optical flow orientations of the new frames. This initialization is also heuristic, and lacks a robust learning framework. While we also set out with the goal of finding objects in motion, our solution to this problem is a novel learning-based method. Scene flow, i.e., 3D motion field in a scene [38], is another form of motion estimation, but is computed with additional information, such as disparity values computed from stereo images [16, 42], or estimated 3D scene models [40]. None of these methods follows a CNN-based learning approach, in contrast to our MP-Net.

Video object segmentation. The task of segmenting objects in video is to associate pixels belonging to a class spatio-temporally; in other words, extract segments that respect object boundaries, as well as associate object pixels temporally whenever they appear in the video. This can be accomplished by propagating manual segment labels in one or more frames to the rest of the video sequence [2,18,25,39]. This class of methods is not applicable to our scenario, where no manual segmentation is available.

Our approach to solve the segmentation problem does not require any manually-marked regions. Several methods in this paradigm generate an over-segmentation of videos [4, 13, 20, 23, 43]. While this can be a useful intermediate step for some recognition tasks in video, it has no notion of objects. Indeed, most of the extracted segments in this case do not directly correspond to objects, making it non-trivial to obtain video object segmentation from this intermediate result. An alternative to this is motion segmentation [5, 11, 28], which produces more coherent regions with point trajectories. They, however, assume homogeneity of motion over the entire object, which is not valid for non-rigid objects.

Another class of segmentation methods cast the problem as a foreground-background classification task [10, 22, 29, 35, 41, 45]. Some of these first estimate a region [29, 41] or regions [22, 45], which potentially correspond(s) to the foreground object, and then learn foreground/background appearance models. The learnt models are then integrated with other cues, e.g., saliency maps [41], pairwise constraints [29, 45], object shape estimates [22], to compute the final object segmentation. Alternatives to this framework have introduced: (i) long-range interactions between distinct parts of the video to overcome noisy initializations in low-quality, unconstrained videos [10], and (ii) occluder/occluded relations to obtain a layered video segmentation [35]. Our proposed method outperforms all the top ones from this class of segmentation approaches, as

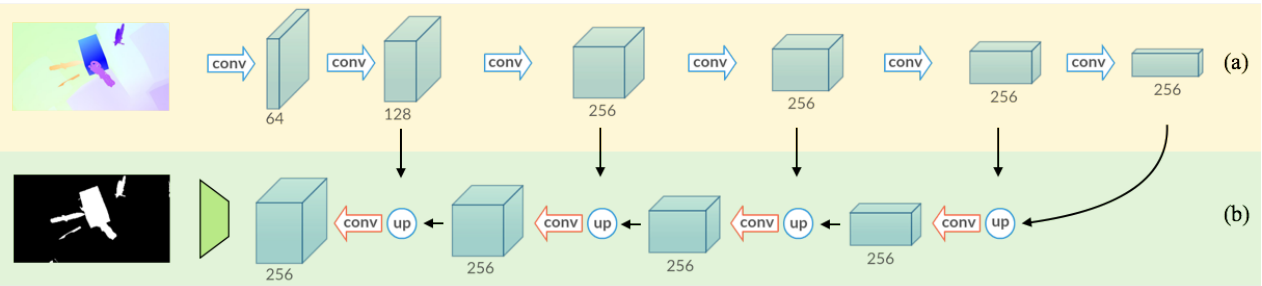


Figure 3. Our motion pattern network: MP-Net. The blue arrows in the encoder part (a) denote convolutional layers, together with ReLU and max-pooling layers. The red arrows in the decoder part (b) are convolutional layers with ReLU, 'up' denotes 2×2 upsampling of the output of the previous unit. The unit shown in green represents bilinear interpolation of the output of the last decoder unit.

shown in Section 6.

Related CNN architectures. Our CNN model predicts labels for every pixel, similar to CNNs for other tasks, such as semantic segmentation [14, 24, 33], optical flow [8] and disparity/depth [9, 26] estimation. We adopt an encoder-decoder style network, inspired by the success of similar architectures in [8, 24, 33]. They first learn a coarse representation with receptive fields of gradually increasing sizes, and then iteratively refine it with upconvolutional layers, i.e., by upsampling the feature maps and performing convolutions, to obtain an output at the original high-resolution. In contrast to [8, 24], which predict labels in each of the upconvolutional layers, we concatenate features computed at different resolutions to form a strong representation, and estimate the labels in the last layer. Our architecture also has fewer channels in the layers in the encoding part, compared to [33], to accomodate larger training-set batches, and thus decrease the training time. More details of our architecture are presented in Section 3.1.

3. Learning Motion Patterns

Our MP-Net takes the optical flow field corresponding to two consecutive frames of a video sequence as input, and produces per-pixel motion labels. In other words, we treat each video as a sequence of frame pairs, and compute the labels independently for each pair. As shown in Figure 3, the network comprises several "encoding" (convolutional and max-pooling) and "decoding" (upsampling and convolutional) layers. The motion labels are produced by the last layer of the network, which are then rescaled to the original image resolution (see §3.1). We train the network entirely on synthetic data—a scenario where ground-truth motion labels can be acquired easily (see §3.2).

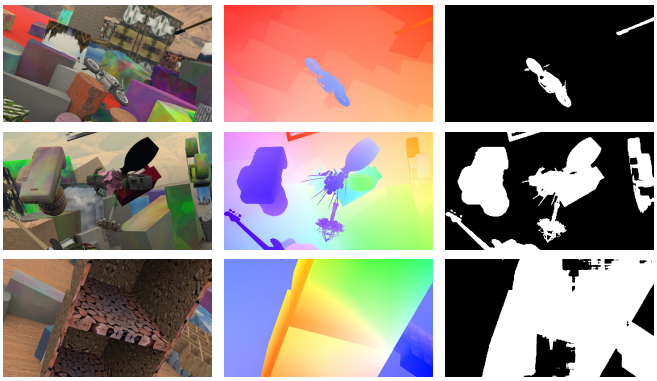
3.1. Network architecture

Our encoder-decoder style network is motivated by the goal of segmenting diverse motion patterns in flow fields, which requires a large receptive field as well as an output at the original image resolution. A large receptive field is critical to incorporate context into the model. For example, when the spatial region of support (for performing convolution) provided by a small receptive field falls entirely within an object with non-zero flow values, it is impossible to determine whether it is due to object or camera motion. On the other hand, a larger receptive field will include regions corresponding to the object as well as background, providing sufficient context to determine what is moving in the scene. The second requirement of output generated at the original image resolution is to capture fine details of objects, e.g., when only a part of the object is moving. Our network satisfies these two requirements with: (i) the encoder part learning features with receptive fields of increasing sizes, and (ii) the decoder part upsampling the intermediate layer outputs to finally predict labels at the full resolution.

Figure 3 illustrates our network architecture. Optical flow field input is processed by the encoding part of the network (denoted by (a) in the figure) to generate a coarse representation that is a 32×32 downsampled version of the input. Each 3D block here represents a feature map produced by a set of layers. In the encoding part, each feature map is a result of applying convolutions, followed by a ReLU non-linearity layer, and then a 2×2 max-pooling layer. The coarse representation learnt by the final set of operations in this part, i.e., the 32×32 downsampled version, is gradually upsampled by the decoder part ((b) in the figure). In each decoder step, we first upsample the output of the previous step by 2×2, and concatenate it with the corresponding intermediate encoded representation (illustrated with black arrows pointing down in the figure). This upscaled feature map is then processed with two convolutional layers, followed by non-linearities, to produce input for the next (higher-resolution) decoding step. The final decoder step produces a motion label map at half the original resolution. We perform a bilinear interpolation on this result to estimate labels at the original resolution.

3.2. Training with synthetic data

We need a large number of fully-labelled examples to train a convolutional network such as the one we propose.



(a) (b) (c)

Figure 4. Each row shows: (a) example frame from a sequence in the FlyingThings3D dataset [26], (b) ground-truth optical flow of (a), which illustrates motion of both foreground objects and background with respect to the next frame, and (c) our estimate of moving objects in this scene with ground-truth optical flow as input.

In our case, this data corresponds to videos of several types of objects, captured under different conditions (e.g., moving or still camera), with their respective moving object annotations. No large dataset of real-world scenes satisfying these requirements is currently available, predominantly due to the cost of generating ground-truth annotations for every frame. We adopt the popular approach of using synthetic datasets, followed in other work [8, 12, 26]. Specifically, we use the FlyingThings3D dataset [26] containing 2250 video sequences of several objects in motion, with ground-truth optical flow. We augment this dataset with ground-truth moving object labels, which are accurately estimated using the disparity values and camera parameters available in the dataset, as outlined in Section 5. See Figure 2 for an illustration.

We train the network with mini-batch SGD under several settings. The one trained with ground-truth optical flow as input shows the best performance. This is analyzed in detail in Section 6.2. Note that, while we use ground-truth flow for training and evaluating the network on synthetic datasets, all our results on real-world test data use only the estimated optical flow. After the convergence of the training procedure, we obtain a learnt model for motion patterns.

Our approach capitalizes on the recent success of CNNs for pixel-level labelling tasks, such as semantic image segmentation, which learn feature representations at multiple scales in the RGB space. The key to their top performance is the ability to capture local patterns in images. Various types of object and camera motions also produce consistent local patterns in the flow field, which our model is able to learn to recognize. This gives us a clear advantage over other pixel-level motion estimation techniques [3, 27] that can not detect local patterns. Motion boundary based heuristics used in [29] can be seen as one particular type of pattern, representing independent object motion. Our model is able to learn many such patterns, which greatly improves the quality and robustness of motion estimation.

4. Detecting Motion Patterns

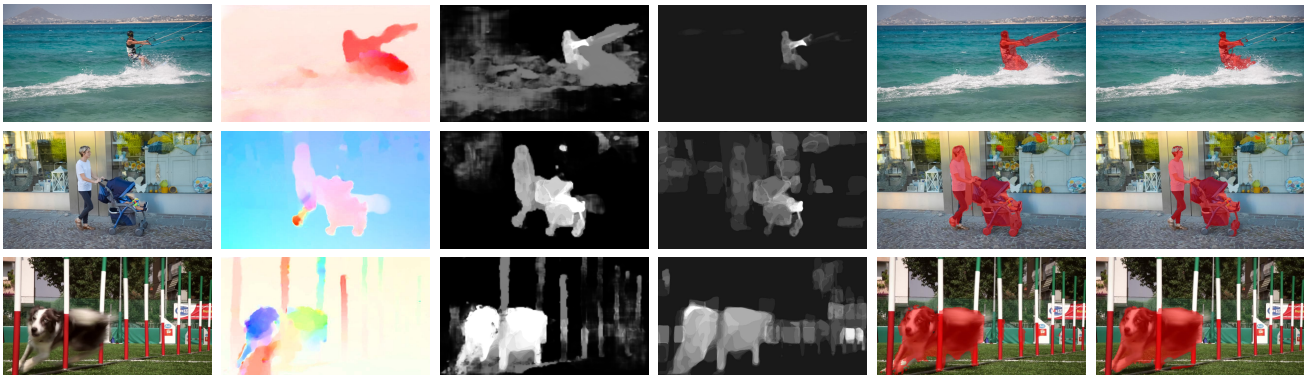
We apply our trained model on synthetic (FlyingThings3D) as well as real-world (DAVIS, BMS-26) test data. Figure 4 shows sample predictions of our model on the FlyingThings3D test set with ground-truth optical flow as input. Examples in the first two rows show that our model accurately identifies fine details in objects: thin structures even when they move very subtlely, such as the neck of the guitar in the top-right corner in the first row (see the subtle motion in the optical flow field (b)), fine structures like leaves in the vase, and the guitar's headstock in the second row. Furthermore, our method successfully handles objects exhibiting highly varying motions in the second example. The third row shows a limiting case, where the receptive field of our network falls entirely within the interior of a large object. Traditional approaches, such as RANSAC, do not work in this case either.

In order to detect motion patterns in real-world videos, we first compute optical flow with popular methods [6, 32]. With this flow as input to the network, we estimate a motion label map, as shown in the examples in Figure 5(c). Although the prediction of our frame-pair feedforward model is accurate in several cases ((c) in rows 2 and 3 in the figure), we are faced with two challenges, which were not observed in the synthetic training set. The first one is motion of *stuff* [1] in a scene, e.g., patterns on the water due to the kiteboarder's motion (first row in the figure), which is irrelevant for moving object segmentation. The second one is significant errors in optical flow, e.g., in front of the pram ((c) in the middle row in the figure). We address these challenges by: (i) incorporating object proposals [31] into our framework, and (ii) refining the result with a fully-connected conditional random field (CRF) [21]. The following two sections present these in detail, and their influence is analyzed in Section 6.3.

4.1. Segmenting real-world videos

As mentioned in the example above (Figure 5, first row), real-world videos may contain *stuff* (water in this case) undergoing independent motion. While it is interesting to study this motion, and indeed, our model estimates it (network prediction (c) in the first row), it is not annotated in any of the standard datasets for moving object segmentation. In order to perform a fair evaluation on standard benchmarks, we introduce the notion of objects, with an objectness score, computed from object proposals, to eliminate "moving stuff." We combine this score with our network output to obtain an updated prediction.

We first generate object proposals in each frame with a state-of-the-art method [31]. We then use a pixel-level



(a) (b) (c) (d) (e) (f)

Figure 5. Sample results on the DAVIS dataset showing all the components of our approach. Each row shows: (a) video frame, (b) optical flow estimated with LDOF [6], (c) output of our MP-Net with LDOF flow as input, (d) objectness map computed with proposals [31], (e) initial moving object segmentation result, (f) segmentation refined with CRF.

voting scheme to compute an objectness score. The score at a pixel i is the number of proposals that include it. This score is normalized by the total number of proposals to obtain o_i, the objectness score at pixel i in the $0-1$ range. In essence, we aggregate several proposals, which are likely to represent objects of interest, to obtain an objectness map, as shown in the examples in Figure 5(d). We then combine this with the motion prediction of our MP-Net at pixel i, $m_i \in [0,1]$, to obtain an updated prediction p_i as: $p_i = \min(m_i * (k + o_i), 1)$, where $k \in [0,1]$ is a parameter controlling the influence of objectness. It is set to 0.5 to ensure that a high confidence network prediction m_i gets suppressed only when there are no objects proposals supporting it. In the example with the kiteboarder (first row in Figure 5), the objectness map (d) has no object proposals on water, the "moving stuff," and eliminates it, to obtain segmentation (e) that corresponds to the moving object.

4.2. Refining the segmentation

As shown on the synthetic test sequences, Figure 4, our model produces accurate object boundaries in several cases. This is in part due to precise optical flow input; recall that we use ground-truth flow on synthetic data. Naturally, computed flow is less accurate than this, and can often fail to provide precise object boundaries (see examples in Figure 5(b)), especially in low-texture regions. Such errors inevitably result in imprecise motion segments. To address this, we follow the common practice of refining segmentation results with a CRF [7]. We use a fully-connected CRF [21], with our predictions updated with objectness scores as the unary terms, and standard colour-based pairwise terms. The refinement is shown qualitatively in Figure 5(f), which improves over the initial segmentation in (e), e.g., contours of the person pushing the pram in the middle row.

5. Datasets

FlyingThings3D (FT3D). We train our network with the synthetic FlyingThings3D dataset [26]. It contains videos of various objects flying along randomized trajectories, in randomly constructed scenes. The video sequences are generated with complex and diverse camera motion, which is also randomized. FT3D comprises 2700 videos, each containing 10 stereo frames. The dataset is split into training and test sets, with 2250 and 450 videos respectively. Ground-truth optical flow, disparity, the intrinsic and extrinsic camera parameters, and object instance segmentation masks are provided for all the videos. No annotation is directly available to distinguish moving objects from stationary ones, which is required to train our network. We extract this from the data provided as follows. With the given camera parameters and the stereo image pair, we first compute the 3D coordinates of all the pixels in a video frame t. Using ground-truth flow between frames t and $t+1$ to find a pair of corresponding pixels, we retrieve their respective 3D scene points. Now, if the pixel has not undergone any independent motion between these two frames, the scene coordinates will be identical (up to small rounding errors). We will make these labels publicly available. Performance on the test set is measured as the standard intersection over union score between the predicted segmentation and the ground-truth masks.

DAVIS. We use the densely annotated video segmentation dataset [30] exclusively for evaluating our approach. DAVIS is a very recent dataset containing 50 full HD videos, featuring diverse types of object and camera motions. It includes challenging examples with occlusion, motion blur and appearance changes. Accurate pixel-level annotations are provided for the moving object in all the video frames. Note that only a single object is annotated in each video, even if there are multiple moving objects

# dec.	Trained on FT3D with ...	FT3D	DAVIS
1	RGB single frame	68.1	12.7
	RGB pair	69.1	16.6
	GT flow	74.5	44.3
	GT angle field	73.1	46.6
	RGB + GT angle field	74.8	39.6
	LDOF angle field	63.2	38.1
4	GT angle field	85.9	52.4

Table 1. Comparing the influence of different input modalities on the FlyingThings3D (FT3D) test set and DAVIS. Performance is shown as mean intersection over union scores. # dec. refers to the number of decoder units in our MP-Net. Ground-truth flow is used for evaluation on FT3D and LDOF flow for DAVIS.

in the scene. We evaluate our method on DAVIS with the three measures used in [30], namely intersection over union for region similarity, F-measure for contour accuracy, and temporal stability for measuring the smoothness of the segmentation over time. We follow the protocol in [30] and downsample the original frames in our experiments.

BMS-26. We also evaluate on sequences from the Berkeley motion segmentation (BMS-26) dataset [5,37]. The original dataset consists of 26 videos with ground-truth object segmentations for a selection of frames. Observing that annotations in some of these videos do not correspond to objects with independent motion, ten of them were excluded in [3]. In order to compare with [3], we follow their experimental protocol, and evaluate on the same subset of BMS-26. Performance is evaluated with F-measure, as done in [3].

6. Experiments and Results

6.1. Implementation details

Training. We use mini-batch SGD with a batch size of 13—the maximum possible due to GPU memory constraints. The network is trained from scratch with learning rate set to 0.003, momentum to 0.9, and weight decay to 0.005. Training is done for 27 epochs, and the learning rate and weight decay are decreased by a factor of 0.1 after every 9 epochs. We downsample the original frames of the FT3D training set by a factor 2, and perform data augmentation by random cropping and mirroring. Batch normalization [17] is applied to all the convolutional layers of the network.

Other details. We perform zero-mean normalization of the flow field vectors, similar to [34]. When using flow angle and magnitude together (which we refer to as flow angle field), we scale the magnitude component, to bring the two channels to the same range. We use 100 proposals in each frame to compute the objectness score (see §4.1). Also, for a fair comparison to other methods on DAVIS, we do not learn the parameters of the fully-connected CRF on this dataset, and instead set them to values used for the related semantic segmentation task [7]. Our model is implemented in the Torch framework. We will make the source code and the trained models available online.

Variant of our method	Flow used	Mean IoU
MP-Net	LDOF	52.4
MP-Net	EpicFlow	56.9
MP-Net + Objectness	LDOF	63.3
MP-Net + Objectness	EpicFlow	64.5
MP-Net + Objectness + CRF	LDOF	69.7
MP-Net + Objectness + CRF	EpicFlow	68.0

Table 2. Performance of our best network (4 decoder units trained on GT angle field) with additional cues (Objectness, CRF) and different flow inputs (LDOF, EpicFlow) on DAVIS.

6.2. Influence of input modalities

We first analyze the influence of different input modalities on training our network. Specifically, we use RGB data (single frame and image pair), optical flow field (ground truth and estimated one), directly as flow vectors, i.e., flow in x and y axes, or as angle field (flow vector angle concatenated with flow magnitude), and a combination of RGB data and flow. These results are presented on the FT3D test set and also on DAVIS, to study how well the observations on synthetic videos transfer to the real-world ones, in Table 1. For computational reasons we train and test with different modalities on a smaller version of our MP-Net, with one decoder unit instead of four. Then we pick the best modality to train and test the full, deeper version of the network.

From Table 1, the performance on DAVIS is lower than on FT3D. This is expected as there is domain change from synthetic to real data, and that we use ground truth optical flow as input for FT3D test data, but estimated flow [6] for DAVIS. As a baseline, we train on single RGB frames ('RGB single frame' in the table). Clearly, no motion patterns can be learned in this case, but the network performs reasonably on FT3D test (68.1), as it learns to correlate object appearance with its motion. This intuition is confirmed by the fact that 'RGB single frame' fails on DAVIS (12.7), where the appearance of objects and background is significantly different from FT3D. MP-Net trained on 'RGB pair', i.e., RGB data of two consecutive frames concatenated, performs slightly better on both FT3D (69.1) and DAVIS (16.6), suggesting that it captures some motion-like information, but continues to rely on appearance, as it does not transfer well to DAVIS.

Training on ground-truth flow vectors corresponding to the image pair ('GT flow') improves the performance on FT3D by 5.4% and on DAVIS significantly (27.7%). This shows that MP-Net learnt on flow from synthetic examples can be transferred to real-world videos. We then experiment with flow angle as part of the input. As discussed in [27], flow orientations are independent of depth from the

Measure		NLC [10]	CVOS [35]	TRC [11]	MSG [5]	KEY [22]	SAL [41]	FST [29]	Ours
$\mathcal{J}$	Mean	64.1	51.4	50.1	54.3	56.9	42.6	57.5	**69.7**
	Recall	73.1	58.1	56.0	63.6	67.1	38.6	65.2	**82.9**
	Decay	8.6	12.7	5.0	**2.8**	7.5	8.4	4.4	5.6
$\mathcal{F}$	Mean	59.3	49.0	47.8	52.5	50.3	38.3	53.6	**66.3**
	Recall	65.8	57.8	51.9	61.3	53.4	26.4	57.9	**78.3**
	Decay	8.6	13.8	6.6	**5.7**	7.9	7.2	6.5	6.7
$\mathcal{T}$	Mean	35.6	24.3	32.7	25.0	**19.0**	60.0	27.6	68.6

Table 3. Comparison to state-of-the-art methods on DAVIS with intersection over union ($\mathcal{J}$), F-measure ($\mathcal{F}$), and temporal stability ($\mathcal{T}$).

Ground truth | Optical flow [6] | FST [29] | NLC [10] | Ours

Figure 6. Qualitative comparison with top-performing methods on DAVIS. Left to right: ground truth, optical flow [6], results of FST [29], NLC [10], and our MP-Net.

camera, unlike flow vectors, when the camera is undergoing only translational motion. Using the ground truth flow angle field (concatenation of flow angles and magnitudes) as input ('GT angle field'), we note a slight decrease in IoU score on FT3D (1.4%), where strong camera rotations are abundant, but in real examples, such motion is usually mild. Hence, 'GT angle field' improves IoU on DAVIS by 2.3%. We use angle field representation in all further experiments.

Using a concatenated flow and RGB representation ('RGB + GT angle field') performs better on FT3D (by 1.7%), but is poorer by 7% on DAVIS, re-confirming our observation that appearance features are not consistent between the two datasets. Finally, training on computed flow [6] ('LDOF angle field') leads to significant drop on both the datasets: 9.9% on FT3D (for testing GT flow is used) and 8.5% on DAVIS, showing the importance of high-quality training data for learning accurate models. The full version of our MP-Net, with 4 decoder units, improves the IoU by 12.8% on FT3D and 5.8% on DAVIS over its shallower one-unit equivalent.

Notice that the performance of our full model on FT3D is excellent, with the remaining errors mostly coming from inherently ambiguous cases like objects moving close to the camera (see third row in Fig. 4), or very strong object/camera motions. On DAVIS the results are considerably lower despite less challenging motions. To investigate to which extent this is due to the errors in the flow estimation we compute LDOF [6] flow on the FT3D test set and evaluate our full model trained on ground-truth flow. We observe a significant drop in performance by 27.2% (from 85.9% to 58.7%). This confirms the impact of the optical flow quality and suggests that improvements in optical flow estimation can increase the performance of our method on real-world videos.

6.3. Evaluation on real videos

We show the performance of our MP-Net on DAVIS in Table 2, along with a study on the influence of additional cues and the flow used. First, we evaluate the importance of the estimated flow quality by comparing EpicFlow [32], a recent method, and LDOF [6], a more classical method. Using EpicFlow, which leverages motion contours, pro-

Measure	CUT [19]	FST [29]	TRC [11]	MTM [44]	CMS [27]	PCM [3]	MP-Net	MP-Net + FST [29]
$\mathcal{F}$	73.0	64.1	72.8	66.0	62.5	**78.2**	71.8	**78.1**

Table 4. Comparison to state-of-the-art methods on BMS-26 used in [3] with F-measure.

Ground truth | Optical flow [6] | FST [29] | PCM [3] | MP-Net + FST [29]

Figure 7. Qualitative comparison on two sample sequences from BMS-26. Left to right: ground truth, optical flow [6], results of FST [29], PCM [3], and our method.

duces more accurate object boundaries, and improves over MP-Net using LDOF by 4.5%. Incorporating objectness cues with our network ('MP-Net + Objectness' in the table), as described in Section 4.1 improves the segmentation results over 'MP-Net' by 10.9% and 7.6% with LDOF and EpicFlow respectively. Refining these segmentation results with a fully-connected CRF ('MP-Net + Objectness + CRF'), as in Section 4.2, further improves the IoU by 6.4% and 3.5% with LDOF and EpicFlow respectively. This refinement has a significant impact when using LDOF flow, as it improves segmentation around object boundaries, as shown in Figure 5. On the other hand, EpicFlow already incorporates motion boundaries, and a CRF refinement on top of the results with this flow has a less-pronounced improvement. The overall method 'MP-Net + Objectness + CRF' performs better with LDOF (69.7) than EpicFlow (68.0). Although EpicFlow has better precision than LDOF around object boundaries, it tends to make larger errors in other regions, which cannot be corrected with a CRF refinement. We thus use LDOF flow in all the experiments on real-world videos.

6.4. Comparison to the state of the art

Table 3 shows comparison with state-of-the-art methods on DAVIS. Our frame-level approach, 'MP-Net + Objectness + CRF' outperforms all the methods significantly, notably by 5.6% on IoU (mean-$\mathcal{J}$) and 7% on F-measure (mean-$\mathcal{F}$) over the top-performing method, NLC [10]. Note that the two top methods, NLC and FST [29] perform a video-level inference by propagating motion labels through the video, unlike our approach using only a pair of video frames at a time. Our network shows the top performance, by a significant margin, with respect to mean and recall on the IoU and F-measure scores. All the methods perform similarly on the decay scores, which quantifies the performance loss/gain over time. As MP-Net does not perform inference at the video level, the temporal stability measure does not provide a fair comparison with top video-level methods. However, we have included it in the table for the sake of completion.

Table 4 shows quantitative comparison on the subset of BMS-26 used in [3]. We observed that objects are annotated in some of the sequences when they do not undergo independent motion. Thus, the results of our MP-Net are not directly comparable to the other methods which use propagation between frame, though they are still on par with many of the previous methods. To account for this mismatch with MP-Net which segments moving objects, we incorporate our frame-level motion estimation results into a state-of-the-art video segmentation method [29]. This is achieved by replacing the location unary scores in [29] with our motion prediction scores after objectness integration. The results (shown as 'MP-Net + FST [29]' in the table) are significantly better than most previous methods, and on par with PCM [3]. In particular, using our motion prediction instead of location score in [29] improves the result by 14%.

Qualitative results. Figure 6 compares our approach qualitatively to the two top-performing methods, FST [29] and NLC [10], on DAVIS. In the first row, FST localizes the moving boat, but its segmentation leaks into the background region around the boat, due to errors in motion tracking. NLC latches onto the moving water, whereas our MP-Net segments the boat accurately. In the second row, NLC and

our method produce a relatively precise segmentation of the scooter and its driver, while FST misses a parts of the moving object, and also bleeds into the background. In the third row, our segmentation result is more precise and complete than both FST and NLC. The last row shows a failure case, where a part of the motorbike is missing due to very imprecise flow estimation.

Figure 7 compares our results on BMS-26 with the top-method on this dataset, PCM [3], and the baseline video-level approach FST [29]. In the first row, FST segments only one of the two moving cars in the foreground, due to very slow motion of the second car. Introducing our motion prediction into FST segments both these cars. This result is comparable to PCM. None of the methods segments the third car in the background however. In the second row, PCM fails to segment the woman, and FST segments only the jacket, but including our motion estimate into FST significantly improves the result. Tracking errors inherent in FST result in the segmentation leaking into the background.

7. Conclusion

This paper introduces a novel approach for learning motion patterns in videos. Its strength is demonstrated for the task of moving object segmentation, where our method outperforms many complex approaches, that rely on engineered features. Future works includes the development of end-to-end trainable solutions in various related domains: video semantic segmentation, action recognition and tracking. We also plan to extend our method with a memory module for video object segmentation.

Acknowledgments. This work was supported in part by the ERC advanced grant ALLEGRO, the MSR-Inria joint project, a Google research award and a Facebook gift. We gratefully acknowledge the support of NVIDIA with the donation of GPUs used for this research.

References

[1] E. H. Adelson. On seeing stuff: The perception of materials by humans and machines. *Proc. SPIE*, 2001. 4

[2] V. Badrinarayanan, F. Galasso, and R. Cipolla. Label propagation in video sequences. In *CVPR*, June 2010. 2

[3] P. Bideau and E. G. Learned-Miller. It's moving! A probabilistic model for causal motion segmentation in moving camera videos. In *ECCV*, 2016. 1, 2, 4, 6, 8, 9

[4] W. Brendel and S. Todorovic. Video object segmentation by tracking regions. In *ICCV*, 2009. 2

[5] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010. 2, 6, 7

[6] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *PAMI*, 2011. 1, 4, 5, 6, 7, 8

[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015. 5, 6

[8] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 1, 3, 4

[9] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 3

[10] A. Faktor and M. Irani. Video segmentation by non-local consensus voting. In *BMVC*, 2014. 2, 7, 8

[11] K. Fragkiadaki, G. Zhang, and J. Shi. Video segmentation by tracing discontinuities in a trajectory embedding. In *CVPR*, 2012. 2, 7, 8

[12] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016. 4

[13] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. In *CVPR*, 2010. 2

[14] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015. 3

[15] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 1981. 1

[16] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 2007. 2

[17] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 6

[18] S. D. Jain and K. Grauman. Supervoxel-consistent foreground propagation in video. In *ECCV*, 2014. 2

[19] M. Keuper, B. Andres, and T. Brox. Motion trajectory segmentation via minimum cost multicuts. In *ICCV*, 2015. 8

[20] A. Khoreva, F. Galasso, M. Hein, and B. Schiele. Classifier based graph construction for video segmentation. In *CVPR*, 2015. 2

[21] P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. In *NIPS*, 2011. 4, 5

[22] Y. J. Lee, J. Kim, and K. Grauman. Key-segments for video object segmentation. In *ICCV*, 2011. 2, 7

[23] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR*, 2011. 2

[24] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 3

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

[26] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 1, 2, 3, 4, 5

[27] M. Narayana, A. R. Hanson, and E. G. Learned-Miller. Coherent motion segmentation in moving camera videos using optical flow orientations. In *ICCV*, 2013. 2, 4, 6, 8

[28] P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *CVPR*, 2012. 2

[29] A. Papazoglou and V. Ferrari. Fast object segmentation in unconstrained video. In *ICCV*, 2013. 1, 2, 4, 7, 8, 9

[30] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 2, 5, 6

[31] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016. 4, 5

[32] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. EpicFlow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, 2015. 4, 7

[33] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 1, 3

[34] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 6

[35] B. Taylor, V. Karasev, and S. Soatto. Causal video object segmentation from persistence of occlusions. In *CVPR*, 2015. 2, 7

[36] P. H. S. Torr. Geometric motion segmentation and model selection. *Phil. Trans. Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 1998. 1, 2

[37] R. Tron and R. Vidal. A benchmark for the comparison of 3-D motion segmentation algorithms. In *CVPR*, 2007. 2, 6

[38] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *PAMI*, 2005. 1, 2

[39] S. Vijayanarasimhan and K. Grauman. Active frame selection for label propagation in videos. In *ECCV*, 2012. 2

[40] C. Vogel, K. Schindler, and S. Roth. 3D scene flow estimation with a piecewise rigid scene model. *IJCV*, 2015. 2

[41] W. Wang, J. Shen, and F. Porikli. Saliency-aware geodesic video object segmentation. In *CVPR*, 2015. 2, 7

[42] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3D motion understanding. *IJCV*, 2011. 2

[43] C. Xu and J. J. Corso. LIBSVX: A supervoxel library and benchmark for early video processing. *IJCV*, 2016. 1, 2

[44] D. Zamalieva, A. Yilmaz, and J. W. Davis. A multi-transformational model for background subtraction with moving cameras. In *ECCV*, 2014. 8

[45] D. Zhang, O. Javed, and M. Shah. Video object segmentation through spatially accurate and temporally dense extraction of primary object regions. In *CVPR*, 2013. 2