Online Learning for Fast Segmentation of Moving Objects

Liam Ellis, Vasileios Zografos
{liam.ellis,vasileios.zografos}@liu.se

CVL, Linköping University, Linköping, Sweden

Abstract. This work addresses the problem of fast, online segmentation of moving objects in video. We pose this as a discriminative online semi-supervised appearance learning task, where supervising labels are autonomously generated by a motion segmentation algorithm. The computational complexity of the approach is significantly reduced by performing learning and classification on oversegmented image regions (superpixels), rather than per pixel. In addition, we further exploit the sparse trajectories from the motion segmentation to obtain a simple model that encodes the spatial properties and location of objects at each frame. Fusing these complementary cues produces good object segmentations at very low computational cost. In contrast to previous work, the proposed approach (1) performs segmentation on-the-fly (allowing for applications where data arrives sequentially), (2) has no prior model of object types or 'objectness', and (3) operates at significantly reduced computational cost. The approach and its ability to learn, disambiguate and segment the moving objects in the scene is evaluated on a number of benchmark video sequences.

1 Introduction

The task of segmenting moving objects in a sequence of images is a fundamental computer vision problem. It's applications include data compression, visual effects, tracking, object and activity recognition, and video annotation and retrieval. This work addresses the problem of fast, online segmentation of moving objects in video. We pose this as a discriminative online semi-supervised appearance learning task, where supervising labels are autonomously generated by a motion segmentation algorithm, applied on sparse point trajectories from interest point tracking. The motion segmentation weakly (and sometimes noisily) labels the moving objects in the scene, and an appearance classifier gradually learns their appearance. In addition to motion (sparse point trajectories) and appearance cues, we also incorporate (1) an efficient shape-location prior and (2) boundary information (from an initial image oversegmentation), to regularize and refine the learning process. Specifically, the learned appearance classifier is given supervising labels from two sources, first from the sparse segmented point trajectories (called the sparse update) and second by sampling the final segmentation obtained after fusing appearance, shape-location and boundary cues (dense update).

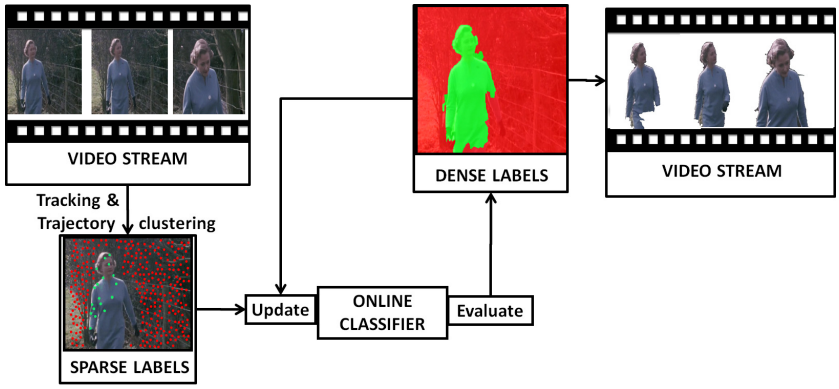


Fig. 1. Basic idea of online semi-supervised learning for segmenting moving objects.

Our method does not have a pre-determined or explicit notion of what an object looks like in the video. Instead its appearance is learned on-the-fly. By avoiding specific assumptions and strict object models, our method is generic. Our main contribution is an autonomous, semi-supervised online learning approach for fast and dense segmentation of moving objects in a video sequence from very sparse trajectories. In contrast to previous work, our proposed method (1) performs segmentation on-the-fly, allowing for applications where the data arrives sequentially; (2) has no prior model of object types or 'objectness' and (3) operates at a significantly reduced computational cost. We analyze the components of our algorithm and experiment on challenging video sequences that demonstrate its performance and utility. An overview of the basic idea of online semi-supervised learning for segmenting moving objects is illustrated in Fig. 1.

2 Related Work

Dense segmentation of moving objects in monocular video sequences is an area that has received considerable attention over the years. Earlier approaches, such as [1–3] rely on the assumption that the background is either stationary or has a simple model, and that only the object of interest is moving. In [4] sparse color features are extracted and a Markov Random Field (MRF) model is used to infuse smoothness for segmentation of videos with little background motion. An improvement to [2] was introduced by [5] using a classifier-based learning approach to deal with a certain degree of background motion and situations where the foreground object is almost stationary. Unlike the proposed approach, all these methods use batch rather than online learning. Solutions such as [6] address the problem of object segmentation by using optical flow for estimating the motion of the scene. Their method, although robust under significant camera motion, cannot handle multiple objects and the optical flow computation makes it unsuitable for real-time applications.

Recently, [7] presented an unsupervised learning approach, which is suited to moving object segmentation when there is very little motion from the object and the camera. Unfortunately, their method is computationally expensive due to the dense optical flow calculation and the mixture of Gaussians (MoG) fitting of the appearance model. In addition, the whole video must be made available for offline batch processing. In [8] Lee et al. extract a series of keyframes in which "object-like" regions are more prominent. They achieve this by using a boundary preserving local feature that has been trained on a database of ground truth object segmentations. Matching these features in the image produces a shape prior that is combined with a basic color MoG model and fused inside a spatio-temporal MRF that is optimized via binary cuts. The method shows promise on realistic datasets, however computationally it is costly; the shape features require extensive training, and computing the prior takes several minutes per frame. Another notable example is the work by [9], on dynamic scenes with rapidly moving objects. Their method clusters pixels together based on multi-scale optical flow, combined with local illumination features in an MRF. Although their method is able to cope very well with highly dynamic objects, (due to motion blur or absence of texture), they can only return a small number of dense segments (object sub-parts) in each image. Brox et al. [10] described an unsupervised method for the segmentation of semi-dense trajectories from the analysis of motion cues over long image sequences. However, they use trajectories provided by [11] which are expensive to obtain. Also, since it relies on 2D motion information, it cannot disambiguate between 3D objects directly, so the authors have to use additional heuristics for merging.

The object segmentation method in [12] builds on previous research from [10]. They use semi-dense motion segmentation to generate a good initial labeling of the moving objects in the scene. Then a very accurate (and computationally expensive) multi-level super-pixel generation method from [13] is carried out to give strong shape priors that preserve the main borders between objects. Super-pixels are labeled and merged using the motion segmentation tracks and a multi-level variational approach. The results presented in [12] are very good and can easily deal with multiple objects. Due to the very costly components, this method is strictly an off-line approach. Even though our solution has certain similarities with the reviewed approaches, it has some unique properties. It is the only approach where a discriminative model of appearance is learned and updated online, supervised by sparse 3D motion features and regularized by shape-location cues and boundary sensitive region segmentations. No costly optimization step is required at the end, and we do not to need to estimate expensive generative models. In this paper, we have restricted the reviewed literature to methods dealing with segmentation of moving objects in video sequences. There is also the related and somewhat overlapping area of dense object segmentation for tracking, with a large number of highly relevant publications. Two notable such examples are the Hough based method by [14] and the older non-parametric approach by [15]. However, dense segmentation methods for tracking applications have different, often more relaxed performance criteria. For example, they tend

to focus on an object centroid or bounding box and not on accurate boundaries that conform to the objects' shapes. Furthermore, they often deal with 2 class foreground/background segmentation and usually require some user-based initialisation. These are the main reasons why we chose to restrict our attention to video object segmentation methods.

3 Our approach

We propose an online approach, where segmentations are made at each frame based only on information from previous frames and a small local temporal window of size N (in our experiments, $N = 10$ frames). For each temporal window, point tracking yields a set of sparse trajectories $\{\mathbf{x}_i^{sparse}\}_{i=1}^{N}$, where $\mathbf{x}_i^{sparse} = \{x_j^1, x_j^2\}_{j=1}^{k}$ is the set of sparse points in the i^{th} frame in the temporal window, and k is the number of trajectories. Next, an unsupervised motion segmentation algorithm assigns labels to trajectories, giving set $\mathbf{S} = \{\mathbf{x}_i^{sparse}, \mathbf{y}_i^{sparse}\}_{i=1}^{N}$, where $\mathbf{y}_i^{sparse} = \{y_j\}_{j=1}^{k}$, is the set of sparse labels in the i^{th} frame in the temporal window, that identifies object assignment. See §3.1 for details.

This set $\mathbf{S}$, of sparse point-label pairs then drives the learning process. Labeled appearance samples (see §3.4) extracted from the point sets in $\mathbf{S}$ provide an initial sparse training set for a discriminative online appearance classifier. Additionally, a basic shape-location model for each object is built directly from $\mathbf{S}$ (see §3.2).

To avoid the high computational complexity of performing learning and classification at the pixel level, and also to incorporate important boundary information, we employ an efficient multi-scale image oversegmentation. Discriminative learning and classification is carried out at the segmented region level, yielding boundary sensitive appearance classifications. (See §3.3).

The fusion of these complementary cues (motion, shape-location and boundary sensitive appearance) yields segmented objects. Segmentations are then used to bootstrap the classifier with dense appearance samples, resulting in greatly improved segmentations in subsequent frames. The cue fusion and semi-supervised online appearance classifier learning algorithm are detailed in §3.5.

3.1 Motion Segmentation

One of the main components of our proposed approach is a fast, unsupervised and non-parametric 3D motion segmentation algorithm. This is used in order to: 1) provide labeled samples for learning the *appearance* for each moving object; 2) provide labeled spatial coordinates for extraction of a basic shape-location model for every object in each frame.

Since we require a motion segmentation algorithm that is both fast and reliable, we choose the LCV method by [16], for which there is available computer code. The LCV method takes as input a point trajectory matrix and outputs a column vector of labels that separate these trajectories into different objects (see Fig. 2(a)). Point trajectories are obtained by first computing a small number

of "Good Features to Track" [17]. These are tracked with a feature tracker [18] using a "track-retrack" scheme, whereby points are tracked from one frame to the next, and back again. Trajectories are rejected if there is sufficient disagreement in their initial and final point locations. Tracking is performed over a small temporal window and only trajectories that persist across the whole window are retained.

The overall process yields an extremely sparse set of reliable labeled point trajectories over each temporal window (approximately 0.05% of image pixels, compared to 3% in [10]). The increased sparsity of points greatly reduces computational cost, but necessitates a semi-supervised learning algorithm that can exploit both labeled data (sparse points) and unlabeled data (all unlabeled pixels).

3.2 Shape-location cues

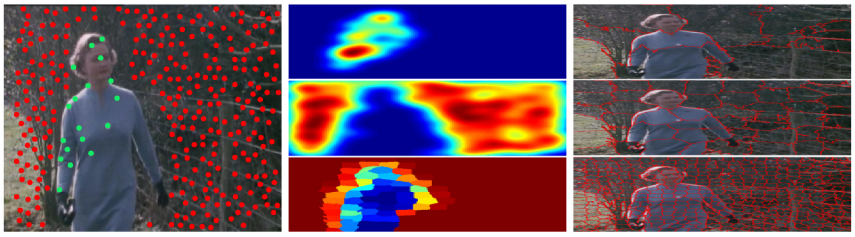
It is important to use a basic shape-location model to weakly regularize the appearance learning process. An estimate of the shape, location and scale of each object is computed in every frame using a Kernel Density Estimate (KDE) [19] based on the sparse point-label pairs output by the motion segmentation, $\{\mathbf{x}^{sparse}, \mathbf{y}^{sparse}\}$. For each object, the 2D spatial distribution is estimated from the sparse point set associated with that object label. The set $\Omega = \{1 : \mathbf{y}_i^{sparse} = \omega\}$ is the set of indices for which the label is ω. The KDE for objects with label ω is defined as:

$$\hat{f}_{\mathbf{H}}(\mathbf{x}) = \frac{1}{\|\Omega\|} \sum_{i \in \Omega} \mathbf{K}_{\mathbf{H}} \left(\mathbf{x} - \mathbf{x}_i\right), \tag{1}$$

and we use a Gaussian kernel with an automatically adapted bandwidth parameter H [20]. The shape KDE is estimated on sparse points and can be sampled densely to obtain a dense confidence map, S_{map}. This model is robust to outliers that may occur due to motion segmentation errors and, in comparison to the shape priors computed in other works (c.f. [8]) it is highly computationally efficient. Additionally, the model is multi-modal, which is necessary for modeling, for instance, the background distribution. See Fig. 2(b) for an example shape-location model.

3.3 Multi-scale Oversegmentation

To avoid the high computational complexity of performing learning and classification at the pixel level, and also to incorporate important boundary information, we employ an efficient multi-scale image oversegmentation. Discriminative learning and classification is then carried out at the segmented region level. Fusing the region classification confidences across all scales yields boundary sensitive per-pixel labeling, and per-pixel classification confidences. By merging classification confidences of oversegmentations at multiple scales, region boundaries that are consistent across many scales are well emphasized, while those boundaries



(a) (b) (c)

Fig. 2. Motion segmentation, shape-location cue and multi-scale oversegmentation. (a) Motion segmentation: sparse points are clustered due to similar trajectories. (b) Shape location cue: top - foreground, middle - background, bottom - one level of multi-scale shape-location cue for background. (c) Three levels of multi-scale oversegmentation.

introduced due to ambiguities and noise are attenuated. Oversegmentation is performed by clustering pixels into superpixels based on their color similarity and proximity in the image plane, as per the SLIC (Simple Linear Iterative Clustering) method, detailed in [21]. Superpixels are computed at M scales, where the finest scale segments the image into M_{MAX} superpixels, and the coarsest scale contains M_{MIN} superpixels. Fig. 2(c) shows three scales of oversegmentation. This step serves a similar purpose to the superpixel step used in [12], however, as is illustrated in Table 1, the segmented regions correspond much less strongly to object boundaries than the superpixels computed in [12]. However, the computational complexity of computing the oversegmentation is considerably lower than for the superpixel computation in [12]. The median of the shape-location KDE within each superpixel is also computed, producing a multi-scale boundary sensitive shape-location confidence, as illustrated in the bottom image of Fig. 2(b).

3.4 Appearance cues

For our proposed on-line segmentation framework, we characterise the visual properties of the image patches with the help of low-level filter systems. We use the so-called dihedral colour filters [22] that meet the requirements of both, fast execution times and simplicity. These filters are constructed with the help of representation theory of discrete groups, which is a generalisation of the theory of the discrete Fourier transform. In this paper, we will use the filters defined on a 5×5 grid around every pixel in a patch. These fitlers represent one special type of edge detectors and in the framework of the representation theory, they transform like the two-dimensional representations of the group. This gives a 75-dimensional feature vector for each pixel. For a detailed description of the group theoretical filter systems we refer the interested reader to [22]. As appearance learning and classification is carried out at the superpixel level, we represent the appearance of each superpixel as the average of the filter responses for all pixels within the superpixel.

3.5 Semi-Supervised Online Appearance Classifier

The unsupervised motion segmentation algorithm (§3.1) provides sparse point-label pairs $\{\mathbf{x}_i^{sparse}, \mathbf{y}_i^{sparse}\}_{i=1}^{N}$ for a semi-supervised online appearance classifier learning algorithm, where N is the number of labeled trajectories. The algorithm uses an Online Random Forest (ORF) classifier as the embedded classifier. Random Forests are an ensemble of randomized decision trees learnt from random bootstrap samples of the original data. In [23] the classifier was implemented as an online learning algorithm. Although experiments with other online classifiers (e.g. online boosting) produce similar results, the ORF is highly computationally efficient both in update and evaluate steps.

There are two classifier update steps in our algorithm, a *sparse update* and a *dense update*. The sparse update is performed at each of the M scales of the oversegmentation, using the training set $\mathbf{A}^{Sparse} = \{\Phi_{i,k}^{sparse}, \mathbf{y}_i^{sparse}\}_{i=1}^{N}$, for k=1 : M, where $\{\Phi_{i,k}^{sparse}\}_{i=1}^{N}$, denotes the appearance features extracted from the regions at scale k containing the points $\{\mathbf{x}_i^{sparse}\}_{i=1}^{N}$. For each scale, the mean appearance of each labeled superpixel (those superpixels containing a tracked point labeled by the motion segmentation) is computed, and the classifier is updated with these labeled samples. Superpixels that contain labels from multiple classes are not used in the update, as they are likely to span multiple objects, these superpixels occur commonly at coarser scales. Also at each scale, all superpixels (including the unlabeled ones) are then evaluated by the classifier yielding multi-scale boundary sensitive appearance confidence maps. For the special case of superpixels that contain multiple different class labels, the confidence is set equal for each class, so that it has no impact on the final merged result. Still at each scale, the multi-scale shape-location and appearance classification confidence maps, (S_{map}^k and A_{map}^k respectively) are fused with a simple linear combination sum[1]:

$$D^k = \alpha \cdot A_{map}^k + (1 - \alpha) \cdot S_{map}^k \tag{2}$$

The next step is to combine the fused confidence maps, D^k, from each scale, k, into a single confidence map. This is computed as the mean, at each pixel, of the fused confidence maps from each scale. $\overline{D} = \frac{1}{M} \sum_{k=1}^{M} D^k$. As $\overline{D}$ is computed as a per-pixel average, and due to the inconsistency of superpixel boundaries across scales, confidence boundaries in $\overline{D}$ are blurred. We therefore compute boundary sensitive version of $\overline{D}$, at the finest superpixel level, by averaging the confidence values of $\overline{D}$ within each superpixel at the finest scale. This results in a confidence map that combines the classification results at all scales, regularized by the shape model, represented at the finest superpixel level. Dense labels, for each superpixel at the finest scale, are finally obtained by thresholding the combined confidence map. The dense update step updates the classifier with the training set $\mathbf{A}^{Dense}$=$\{\Phi_i^{dense}, \mathbf{y}_i^{dense}\}_{i=1}^{M_{MAX}}$, where the dense labels are obtained from the final segmentation result from each frame. M_{MAX} is the number of

[1] All parameters settings are detailed in section §4

Algorithm 1 Online Moving Object Segmentation

Inputs: Sequence of images
for all Temporal windows of N frames **do**
  Track: Obtain point trajectories, $\mathbf{x}_i^{sparse}$
  Motion Segment: Obtain trajectory labels, $\mathbf{y}_i^{sparse}$
  Over Segment: Obtain multi-scale superpixels.
  Sample Appearance: Sample labeled superpixels at each scale
  Sparse Update: Update classifier from $\mathbf{A}^{Sparse}$
  for $i = 1$ to N **do**
    for $k = 1$ to M **do**
      Evaluate Classifier: Obtain appearance confidence map, A_{map}^k
      Shape and Location: Obtain shape and location confidence maps, S_{map}^k
      Cue Fusion: As in Eq. 2
    end for
    Combine Scales: Compute per pixel average of fused confidence maps $\overline{D} = \frac{1}{M}\sum_{k=1}^{M} D^k$
    Compute Superpixel Confidence Map: Compute average confidence values of $\overline{D}$ within each superpixel at the finest scale
    Threshold: Threshold confidence map to obtain labels for each superpixel at finest scale
    Dense Update: Update classifier from $\mathbf{A}^{Dense}$
  end for
end for
Outputs: Moving object segmentations in all images

superpixels at the finest scale, and $\{\Phi_i^{dense}\}_{i=1}^{M_{MAX}}$ are the mean appearance features in each superpixel at the finest scale. If the update were performed prior to the fusion step, the algorithm would be self-training i.e. a 'single-view weakly supervised algorithm' [24]. Such algorithms can suffer from reinforcing feedback errors, i.e. initial misclassifications are included in the bootstrap training data, resulting in ever increasing classification errors. By updating the classifier with the post-fusion segmentation result, the learning process utilizes multiple views (appearance, shape) of the data. This multi-view approach constrains the learning process by only bootstrapping the classifier with data where there is agreement between the views. The complete online segmentation algorithm is detailed in Algorithm 1.

4 Implementation Details

All timings quoted are based on a 64bit, 2.83 GHz Quad Core CPU with 4GB RAM.

Tracking: Tracking over a 10 frame temporal window of 640×480 images takes approximately 0.2 sec and, depending on scene content, returns between 100 and 1000 trajectories. A track-retrack threshold of 0.3 pixels is used to reject unreliable trajectories.

Motion segmentation: The parameter values used are the same as detailed in the original paper [16]. The only difference being that the method is applied to shorter trajectories. Motion segmentation takes approximately 0.02 sec for 1000 trajectories over 5 frames.

ORF classifier: For all experiments, the ORF contains 200 trees with a maximum depth of 50. During update, 200 random tests are evaluated at each node split. Lower values for all these parameters will tend to reduce both performance and computational cost. However varying these parameters in the range (200 ± 100 for number of trees, 50 ± 30 for max tree depth, 200 ± 100 for number of splits) has little effect on the performance of the algorithm, and as the classifier is computationally inexpensive, the exact parameter settings are not critical. The ORF performed best when the update samples were interleaved (e.g. alternating labels for a two class case), and also balanced (equal number of samples from each class). To achieve this, and control computational cost, 500 samples from each class are randomly sampled (with replacement) from the training data at both sparse and dense updates. Updating with 1000 appearance samples (75 dimensions) takes approximately 1 sec. Classifying a 640×480 image (307200 75D samples) takes approximately 4 sec.

Cue fusion: For merging the shape and appearance confidence maps, as in equ. 2, $\alpha = 0.7$ for all experiments.

5 Experiments

We evaluate the proposed approach on a number of benchmark sequences, from Berkeley Motion Segmentation Dataset [10] and SegTrack [25].

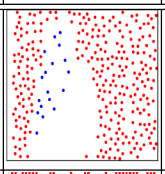
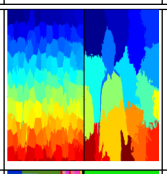
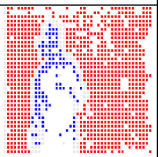
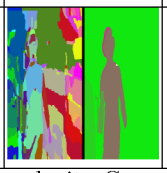
	Trajectories	Result	S.pixels	Result	Object seg.	Result
Our	0.13 sec		2.1 sec		16.61 sec	
[12]	163.37 sec		274.8 sec		33.72 sec	

Table 1. Process Complexity Comparison

First, we compare the computational complexity (in terms of run time on a 64bit, 2.83 GHz Quad Core CPU with 4GB RAM machine) of the key processes

in our method with that of a state-of-the-art method [12]. Table 1 shows both the runtime, and the result of the three main processes: (1) Segmented Trajectories (includes interest point tracking and clustering trajectories over a 10 frame window), (2) oversegmentation/superpixel computation, (3) object segmentation (for both methods, this includes all processes not carried out in process (1) or (2)). As can be seen, the computational cost for all the processes of our method are considerably lower than for [12]. It is also evident that both the segmented trajectories, and the superpixels used by [12] are closer to the final solution than our results. In fact the coarsest level superpixels obtained by [13] are already very close to the final solution. Note that the final segmentation results are for the 1st frame of the sequence, and due to the continuing updates of the appearance classifier, results on later images in the sequence are generally improved for our method. The times quoted are for the images shown, but are representative of all tested sequences.

Fig. 3 shows the temporal evolution of object segmentations for three challenging sequences from the Berkeley Motion Segmentation Dataset [10]. It can be seen that, given a very poor initial discrimination between object and background appearance, by the end of the sequences, the classifier has learnt to effectively discriminate the object.

Fig. 3. Temporal evolution of object segmentations that illustrate the online learning aspect of the method. The three rows contain the Marple7, Marple11 and Tennis sequences respectively from [10].

Figs. 4, 5 and 6 show qualitative and quantitative comparisons to the variational approach of Ochs & Brox [12]. For each sequence, a labeling error is computed on each ground-truth frame. The error is an object size normalised sum of misclassified pixels. The normalization is obtained by dividing the misclassified pixel sum for each method, by the misclassified pixel sum of a trivial segmentation, that labels all pixels with one label.

Fig. 4 shows results on the parachute sequence. Fine details and object boarders are less accurate due to the higher accuracy of [13] over [21], resulting in a

small error bias. However, the stability of the learning approach is demonstrated by the improved segmentation (over [12]), as evident from frame 40 onwards, where [12] starts to segment only part of the parachute.

To compare the performance of our approach, without the bias introduced by using different motion segmentations and superpixels, we evaluate our method using trajectories from [10] and superpixels from [13] (as in [12]). Fig. 5 shows: (top row) input images for which ground-truth is available; (2nd row) the segmentation result of [12]; (3rd row) the segmentation result of our method using [10] and [13]; (4th row) the segmentation result of our method as described. Between frame 50 and 100 the target stops moving so no short term trajectories are created. However, the proposed learning algorithm performs well when using the long term trajectories of [10], even learning to segment a second person in the scene, where [12] fails.

Fig. 6 presents more comparative results and also the final combined confidence map for each ground-truth frame. The map illustrates the increasing confidence of the classifier as more data arrives.

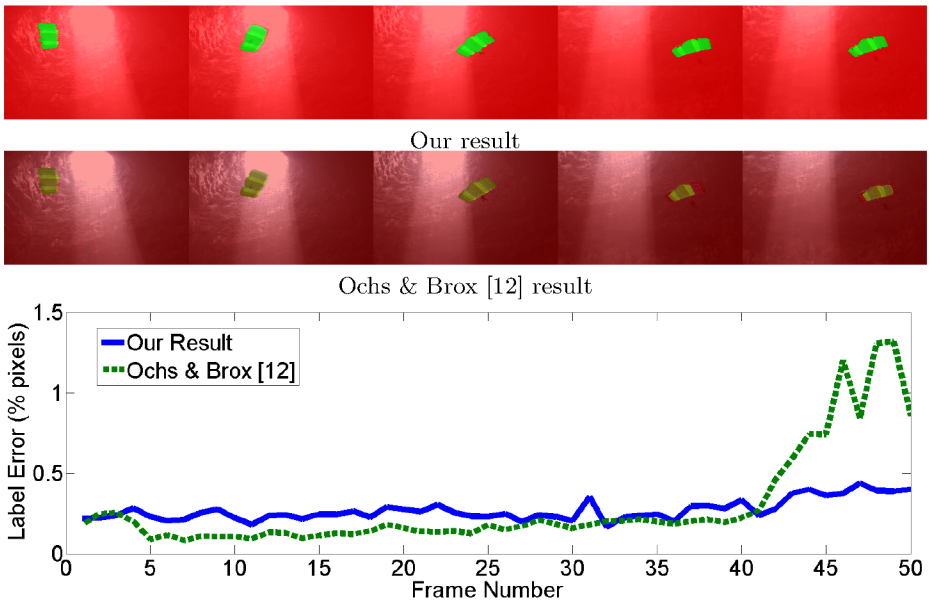


Fig. 4. Comparison of our approach and the variational approach of Ochs & Brox [12], on frames 1, 15, 30, 45 and 50 of the 50 frame parachute sequence from the SegTrack [25] dataset.

Fig. 7 shows some additional examples that illustrate the final segmentation results of our method on video sequences. Note that the colouring of the labels (red, blue) is assigned arbitrarily, so that sometimes background is blue and sometime red.

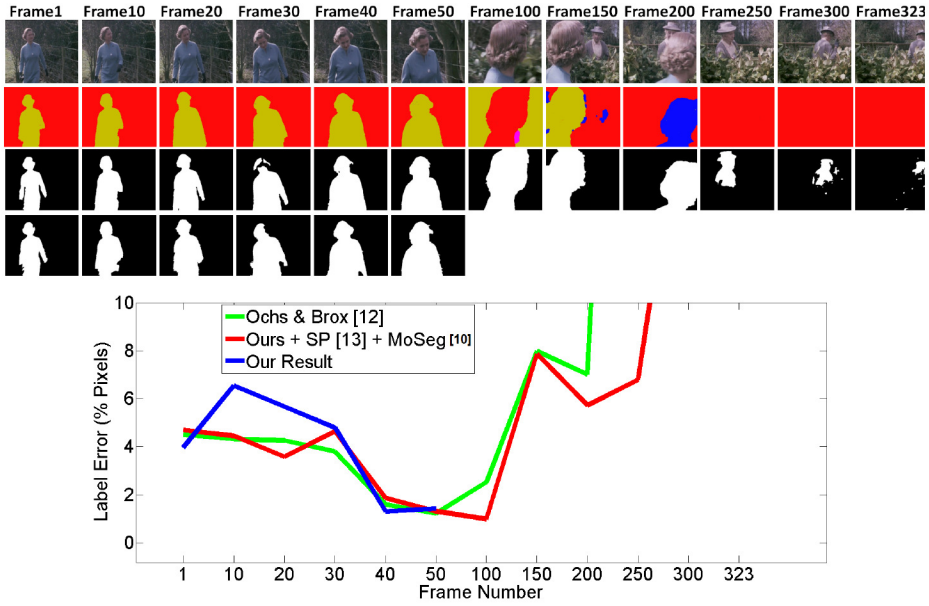


Fig. 5. Comparison with [12] and our method using [10] and [13].

6 Conclusion

In this paper, we have presented a novel online approach for the autonomous segmentation of category-independent moving objects from unannotated video. Given weak sparse supervision, obtained from an unsupervised motion segmentation algorithm, the approach is able to discriminate the appearance of objects. This is achieved by employing a multi-view semi-supervised learning algorithm that combines appearance and shape cues. It is shown that the appearance learning converges, leading to good quality object segmentations.

The learning approach demonstrates good performance despite using computationally inexpensive components (trajectories, superpixels, ORFs), which means that the computational cost of the approach is considerably lower than many existing algorithms, making it possible to simultaneously segment the object and learn the appearance at near real-time.

References

1. Sun, J., Zhang, W., Tang, X., yeung Shum, H.: Background cut. In: ECCV. (2006) 628–641
2. Criminisi, A., Cross, G., Blake, A., Kolmogorov, V.: Bilayer segmentation of live video. In: CVPR. Volume 1. (2006) 53 – 60
3. Monnet, A., Mittal, A., Paragios, N., Ramesh, V.: Background modeling and subtraction of dynamic scenes. In: ICCV. Volume 2. (2003) 1305 –1312

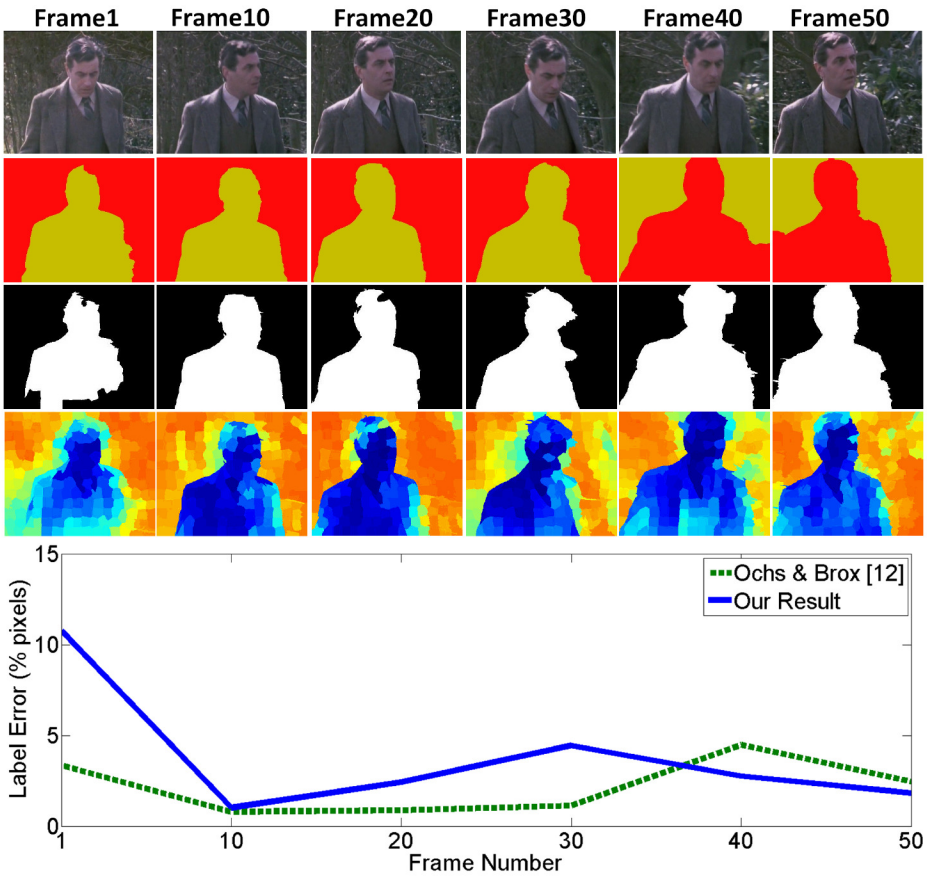


Fig. 6. Comparison with [12], including final confidence maps demonstrating online classifier learning.

4. Han, M., Xu, W., Gong, Y.: Video object segmentation by motion-based sequential feature clustering. In: ACM Multimedia. (2006) 773–782
5. Yin, P., Criminisi, A., Winn, J., Essa, M.: Tree-based classifiers for bilayer video segmentation. In: CVPR. (2007) 1–8
6. Zhang, G., Jia, J., Xiong, W., Wong, T.T., Heng, P.A., Bao, H.: Moving object extraction with a hand-held camera. In: ICCV. (2007) 1 –8
7. F., Gleicher, M.: Learning color and locality cues for moving object detection and segmentation. CVPR (2009) 320–327
8. Lee, Y.J., Kim, J., Grauman, K.: Key-segments for video object segmentation. In: ICCV. (2011)
9. Bugeau, A., Perez, P.: Detection and segmentation of moving objects in highly dynamic scenes. In: CVPR. (2007)
10. Brox, T., Malik, J.: Object segmentation by long term analysis of point trajectories. In: ECCV. (2010) 282–295

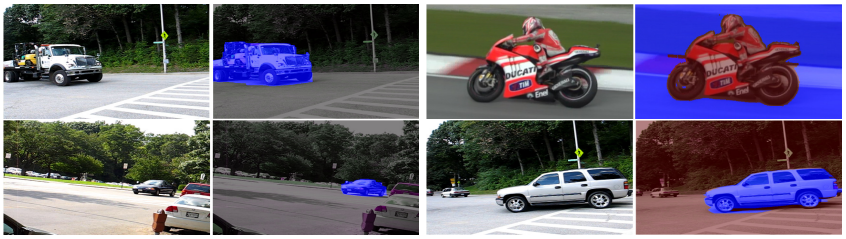


Fig. 7. Some additional examples that illustrate the final segmentation results of our method on video sequences.

11. Brox, T., Malik, J.: Large displacement optical flow: Descriptor matching in variational motion estimation. IEEE PAMI **33** (2011) 500 –513
12. Ochs, P., Brox, T.: Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In: ICCV. (2011)
13. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE PAMI **33** (2011) 898–916
14. Godec, M., Roth, P., Bischof, H.: Hough-based tracking of non-rigid objects. In: ICCV. (2011) 81 –88
15. Lu, L., Hager, G.D.: A nonparametric treatment on location/segmentation based visual tracking. In: CVPR. (2007)
16. Zografos, V., Nordberg, K.: Fast and accurate motion segmentation using linear combination of views. In: BMVC. (2011) 12.1–12.11
17. Shi, J., Tomasi, C.: Good features to track. In: CVPR. (1994) 593 –600
18. Lucas, B.D., .Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Int. joint conference on A.I. Volume 2. (1981) 674–679
19. Hwang, J.N., Lay, S.R., Lippman, A.: Nonparametric multivariate density estimation: A comparative study. IEEE Trans. Signal Processing **42** (1994) 2795–2810
20. Botev, Z.I., Grotowski, J.F., Kroese, D.P.: Kernel density estimation via diffusion. Annals of Statistics (2010)
21. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Ssstrunk, S.: SLIC Superpixels Compared to State-of-the-art Superpixel Methods. IEEE PAMI (2012)
22. Lenz, R., Bui, H.T., Takase, K.: A group theoretical toolbox for color image operators. In: ICIP. Volume 3. (2005) 557–560
23. Saffari, A., Leistner, C., Santner, J., Godec, M., Bischof, H.: On-line Random Forests. In: ICCV Workshops, IEEE (2009) 1393–1400
24. Ng, V., Cardie, C.: Weakly supervised natural language learning without redundant views. In: NAACL. Volume 1. (2003) 94–101
25. Tsai, D., Flagg, M., M.Rehg, J.: Motion coherent tracking with multi-label mrf optimization. BMVC (2010)