Computational Visual Media
https://doi.org/10.1007/s41095-019-0145-0 Vol. 5, No. 3, September 2019, 267–291
Research Article

Object removal from complex videos using a few annotations

Thuc Trinh Le[1] (✉), **Andrés Almansa**[2], **Yann Gousseau**[1], **and Simon Masnou**[3]

Abstract We present a system for the removal of objects from videos. As input, the system only needs a user to draw a few strokes on the first frame, roughly delimiting the objects to be removed. To the best of our knowledge, this is the first system allowing the semi-automatic removal of objects from videos with complex backgrounds. The key steps of our system are the following: after initialization, segmentation masks are first refined and then automatically propagated through the video. Missing regions are then synthesized using video inpainting techniques. Our system can deal with multiple, possibly crossing objects, with complex motions, and with dynamic textures. This results in a computational tool that can alleviate tedious manual operations for editing high-quality videos.

Keywords object removal; object segmentation; object tracking; video inpainting; video completion

1 Introduction

In this paper, we propose a system to remove one or more objects from a video, starting with only a few user annotations. More precisely, the user only needs to approximately delimit in the first frame the objects to be edited. Then, these annotations are refined and propagated through the video. One or more objects can then be removed automatically. This results in a flexible computational video editing tool, with numerous potential applications. Removing unwanted objects (such as a boom microphone) or people (such as an unwanted wanderer) is a common task in video post-production. Such tasks are critical given the time constraints of movie production and the prohibitive costs of reshooting complex scenes. They are usually achieved through extremely tedious and time-consuming frame-by-frame processes, for instance using the Rotobrush tool from Adobe After Effects [1] or professional visual effects software such as SilhouetteFX or Mocha. More generally, the proposed system paves the way to sophisticated movie editing tasks, ranging from crowd suppression to unphysical scene modification, and has potential applications for multi-layered video editing.

Two main challenges arise in developing such a system. Firstly, *no* part of the objects to be edited should remain in the tracking part of the algorithm; otherwise, they would be propagated and enlarged by the completion step, resulting in unpleasant artifacts. Secondly, the human visual system is good at spotting temporal discontinuities and aberrations, making the completion step a tough one. We address both issues in this work.

The first step of our system consists of transforming a rough user annotation into a mask that accurately represents the object to be edited. For this, we use a classical strategy relying on a CNN-based edge detector, followed by a watershed transform yielding super-pixels, which are eventually selected by the user to refine the segmentation mask. After this step, a label is then given to each object. The second step is the temporal propagation of the labels. There we make use of state-of-the-art advances in CNN-based multiple object segmentation. Furthermore, our approach includes an original and crucial algorithmic block which consists in learning the transition zones

1 LTCI, Télécom ParisTech, Université Paris-Saclay, 75013 Paris, France. E-mail: T. T. Le, thuc.le@telecom-paristech.fr (✉); Y. Gousseau, yann.gousseau@telecom-paristech.fr.
2 MAP5, CNRS & Université Paris Descartes, 75006 Paris, France. E-mail: andres.almansa@parisdescartes.fr.
3 Univ Lyon, Université Claude Bernard Lyon 1, CNRS UMR 5208, Institut Camille Jordan, 69622 Villeurbanne, France. E-mail: masnou@math.univ-lyon1.fr.
Manuscript received: 2019-03-20; accepted: 2019-05-18

between objects and the background, in such a way that objects are fully covered by the propagated masks. We call the resulting block a *smart dilation* by analogy with the dilation operators of mathematical morphology. Our last step is then to remove some or all of the objects from the video, depending on the user's choice. For this, we employ two strategies: motion-based pixel propagation for the static background, and patch-based video completion for dynamic textures. Both methods rely heavily on the knowledge of segmented objects. This interplay between object segmentation and the completion scheme improves the method in many ways: it allows for better video stabilization, for faster and more accurate search for similar patches, and for more accurate foreground–background separation. These improvements yield completion results with very little or no temporal incoherence.

We illustrate the effectiveness of our system through several challenging cases including severe camera shake, complex and fast object motions, crossing objects, and dynamic textures. We evaluate our method on various datasets, for both object segmentation and object removal. Moreover, we show on several examples that our system yields comparable or better results than state-of-the-art video completion methods applied to manually segmented masks.

This paper is organized as follows. First, we briefly explore some related works (Section 2). Next, we introduce our proposed approach which includes three steps: first-frame annotation, object segmentation, and object removal (Section 3). Finally, we give experimental results as well as an evaluation and comparison with other state-of-the-art methods. A shorter version of this work can be found in Ref. [2].

2 Related works

The proposed computational editing approach is related to several families of works that we now briefly review.

2.1 Video object segmentation

Video object segmentation, the process of extracting space–time segments corresponding to objects, is a widely studied topic whose complete review is beyond the scope of this paper. For a long time, such methods were not accurate enough to avoid using green-screen compositing to extract objects from video. Significant progress was achieved by the end of the 2000s for supervised segmentation: see e.g., Ref. [1]. In particular, the use of supervoxels became the most flexible way to incorporate user annotations in the segmentation process [3, 4]. Other efficient approaches to the supervised object segmentation problem are introduced in Refs. [5, 6].

A real breakthrough occurred with approaches relying on convolutional neural networks (CNNs). In the DAVIS-2016 challenge [7], the most efficient methods were all CNN-based, both for unsupervised and semi-supervised tasks. For the semi-supervised task, where a first frame annotation is available, methods mostly differ in the way they train the networks. The one shot video object segmentation (OSVOS) method, introduced in Ref. [8], starts from a pre-trained network and retrains it using a large video dataset, before fine-tuning it per-video using annotation on the first frame to focus on the object being segmented. With a similar approach, Ref. [9] relies on an additional mask layer to guide the network. The method in Ref. [10] further improves the results from OSVOS with the help of a multi network cascade (MNC) [11].

All these approaches work image-per-image without explicitly checking for temporal coherence, and therefore can deal with large displacements and occlusions. However, since their backbone is a network used for semantic segmentation, they cannot distinguish between instances of the same class or between objects that resemble each other.

Another family of works deals with the segmentation of multiple objects. Compared with the single object segmentation problem, an additional difficulty here is to distinguish between different object instances which may have similar colors and may cross each other. Classical approaches include graph-based segmentation using color or motion information [12–14], the tracking of segmentation proposals [15, 16], or bounding box guided segmentation [17, 18].

The DAVIS-2017 challenge [19] established a ranking between methods aiming at semi-supervised segmentation of multiple objects. Again, the most efficient methods were CNN-based. It is proposed in Ref. [20] to modify the OSVOS network [8] to work with multiple labels and to perform online fine-tuning

to boost performance. In Ref. [21], the networks introduced in Ref. [22] are adapted to the purpose of multiple object segmentation through the heavy use of data augmentation, still using annotation of the first frame. The authors of this work also exploit motion information by adding optical flow information to the network. This method is further improved in Ref. [23] by using a deeper architecture and a re-identification module to avoid propagating errors. This last method has achieved the best performance in the DAVIS-2017 challenge [19]. With a different approach, Hu et al. [24] employ a recurrent network exploiting long-term temporal information.

Recently, with the release of a large-scale video object segmentation dataset for the YouTube video object segmentation (YouTube-VOS) challenge [25], many further improvements have been made in the field. Among them, one of the most notable is PreMVOS [26] which has won the 2018 DAVIS challenge [27] and the YouTube-VOS challenge [25].

In PreMVOS, the algorithm first generates a set of accurate segmentation mask proposals for all objects in each frame of a video. To achieve this, a variant of the mask R-CNN [28] object detector is used to generate coarse object proposals, and then a fully convolutional refinement network inspired by Ref. [29] and based on the DeepLabv3+ [30] architecture produces accurate pixel masks for each proposal. Secondly, these proposals are selected and merged into accurate and temporally consistent pixel-wise object tracks over the video sequence. In contrast with PreMVOS which focuses on accuracy, some methods trade off accuracy for speed. Those methods take the first frame with its mask annotation either as guidance to slightly adjust parameters of the segmentation model [31] or as a reference for segmenting the following frames without tuning the segmentation model [32–34].

Although these methods yield impressive results in terms of the accuracy of the segmentation, they may not be the optimal solutions for the problem we consider in this paper. As noted above, when removing objects from video, it is crucial for the video completion step that no part of the removed objects remains after segmentation. Said differently, we are in a context where *recall* is much more important than *precision*; see Section 4.2 for definitions of these metrics. In the experiments section, we compare our segmentation approach to several state-of-the-art methods with the aim of optimizing a criterion which penalizes under-detection of objects.

2.2 Video editing

Recently, advances in both analysis and processing of video have permitted advances in the emerging field of computational video editing. Examples include, among others, tools for the automatic, dialogue-driven selection of scenes [35], time slice video synthesis [36], and methods for the separate editing of reflectance and illumination components [37]. It is proposed in Ref. [38] to accurately identify the background in video as a basis for stabilization, background suppression, or multi-layered editing. In a sense, our work is more challenging since we need to identify moving objects with enough accuracy that they can be removed seamlessly.

Because we learn a transition zone between objects and the background, our work is also related to image matting techniques [39], and their extension to video [40] as a necessary first step for editing and compositing tasks. Lastly, since we deal with semantic segmentation and multiple objects, our work is also related to the soft semantic segmentation recently introduced for still images [41].

2.3 Video inpainting

Image inpainting, also called image completion, refers to the task of reconstructing missing or damaged image regions by taking advantage of image content outside these missing regions.

The first approaches were variational [42], or PDE-based [43] and dedicated to the preservation of geometry. They were followed by patch-based methods [44, 45], inherited from texture synthesis methods [46]. Some of these methods have been adapted to video, often by mixing pixel-based approaches for reconstructing the background and greedy patch-based strategies for moving objects [47, 48]. In the same vein, different methods have been proposed to improve or speed up reconstruction of the background [49, 50], with the strong limitation that the background should be static. Other methods yield excellent results in restricted cases, such as the reconstruction of cyclic motions [51].

Another family of works which performs very well when the background is static relies on motion-based pixel propagation. The idea is to first infer a motion

field outside and inside the missing regions. Using the completed motion field, pixel values from outside the missing region are then propagated inside it. For example, Grossauer describes in Ref. [52] a method for removing blotches and scratches in old movies using optical flow. A limitation of this work is that the estimation of the optical flow suffers from the presence of the scratches. Using a similar idea, but avoiding calculating the optical flow directly in the missing regions, several methods try to restore the motion field inside these missing regions by gradually propagating motion vectors [53], by sampling spatial-temporal motion patches [54, 55], or by interpolating the missing motion [56, 57].

In parallel, it was proposed in Ref. [58] to address the video inpainting problem as a global patch-based optimization problem, yielding unprecedented time coherence at the expense of very heavy computational costs. The method in Ref. [59] was developed from this seminal contribution, by accelerating the process and taking care of dynamic texture reconstruction. Other state-of-the-art strategies rely on a global optimization procedure, taking advantage of either shift-maps [60] or an explicit flow field [61]. This last method arguably has the best results in terms of temporal coherence, but since it relies on two-dimensional patches, it is unsuitable for the reconstruction of dynamic backgrounds. Recently, it was proposed in Ref. [62] to improve the global strategy of Ref. [59] by incorporating optical flow in a systematic way. This approach has the ability to reconstruct complex motions as well as dynamic textures.

Let us add that the most recent approaches to image inpainting rely on convolutional neural networks and have the ability to infer elements that are not present in the image at hand [63–65]. To the best of our knowledge, such approaches have not been adapted to video because their training cost is prohibitive.

In this work, we propose two complementary ways to perform the inpainting step needed to remove objects in video. The first method is fast and relies on frame-by-frame completion of the optical flow, followed by propagation of voxel values. This approach is inspired by the recently introduced method in Ref. [57], itself sharing ideas with the approach from Ref. [61] and yielding impressive speed gains. Such approaches are computationally efficient but unable to deal with moving backgrounds and dynamic textures. For these complex cases, we rely on a more sophisticated (and much slower) second approach, extending ideas we initially developed in Ref. [62].

3 Proposed method

The general steps of our method are as follows:

(a) First, the user draws a rough outline of each object of interest in one or several frames, for instance in the first frame (see Section 3.1).

(b) These approximate outlines are refined by the system, then propagated to all remaining frames using different labels for different objects (see Section 3.2).

(c) If errors are detected, the user may manually correct them in one or several frames (using step (a)) and propagate these edits to the other frames (using step (b)).

(d) Finally, the user selects which of the selected objects should be removed, and the system removes the corresponding regions from the whole video, reconstructing the missing parts in a plausible way (see Section 3.3). For this last step two options are available: a fast one for static backgrounds, and a more involved one for dynamic backgrounds.

In the first step most methods only select the object to be removed. There are, however, several advantages to tracking multiple objects with different labels:

1. It gives more freedom to the user for the inpainting step with the possibility to produce various results depending on which objects are removed; in addition, objects which are labeled but not removed are considered as important by the system and therefore better preserved during inpainting of other objects.
2. It may produce better segmentation results than tracking a single object, in particular when several objects have similar appearance.
3. It facilitates video stabilization and therefore increases temporal coherence during the inpainting step, as shown in the results (see Section 4.3).

4. It is of interest for other applications, e.g., action recognition or scene analysis.

An illustration of these steps can be found in the supplementary website:
`https://object-removal.telecom-paristech.fr/`.

3.1 First frame annotation

A classical method to cut out an object from a frame involves commercial tools such as the Magic Wand of Adobe Photoshop which are fast and convenient. However, this classical method requires many refinement steps and is not accurate with complex objects. To increase the precision and reduce user interaction, many methods have been proposed where interactive image segmentation is performed using scribbles, point clicks, superpixels, etc. Among them, some state-of-the-art annotators achieve a high degree of precision by using edge detectors to find the contour map and create a set of object proposals from this map [66]; the appropriate regions are then selected by the user using point clicks. The main drawbacks of these approaches are large computation time and a weak level of user input.

In order to balance human effort and accuracy, we adopt a fast and simple algorithm. Our system first generates a set of superpixels from the first image, and then the user can select suitable superpixels by simply drawing a coarse contour around each object. The set of superpixels is created using an edge-based approach. More precisely, the FCN-based edge detector network introduced in Ref. [67] is applied to the first image, and its output is a probability map of edges. Superpixels are extracted from this map by the well-known watershed transform [68], which runs directly on edge scores. There are two main advantages of using this CNN-based method to compute the edge map:

1. It has shown superior performance over traditional boundary detection methods that use local features such as colors and depths. In particular, it is much more accurate.
2. It is extremely fast: one forward pass of the network takes about 2 ms, so the annotation step can be performed interactively in real time.

After computing all superpixels, the user selects the suitable ones by drawing a contour around each target object to get rough masks. Superpixels which overlap these masks by more than 80% are selected. The user can refine the mask by adding or removing superpixels using mouse clicks. As a result, accurate masks for all objects of interest are extracted in a frame after a few seconds of interactive annotation.

3.2 Object segmentation

In this step, we start from the object masks computed on the first frame using the method described in the previous section, and we aim to infer a full space–time segmentation of each object of interest in the whole video. We want our segmentation to be as accurate as possible, in particular without false negatives.

Doing this in complex videos with several objects which occlude each other is an extremely challenging task. As described in Section 2, CNNs have made important breakthroughs in semantic image segmentation with extensions to video segmentation in the last two years [19, 27, 69]. However, current CNN-based semantic segmentation algorithms are still essentially image-based, and do not take global motion information sufficiently into account. As a consequence, semantic segmentation algorithms cannot deal with sequences where: (i) several instances of similar objects need to be distinguished; and (ii) these objects may eventually cross each other. Examples of such sequences are *Les Loulous* [①] introduced in Ref. [59], and *Museum* and *Granados-S3* [②] introduced in Refs. [49, 60].

On the other hand, more classical video tracking techniques like optical flow-based propagation or global graph-based optimization do take global motion information into account [70]. Nevertheless, they are most often based on bounding boxes or rough descriptors and do not provide a precise delineation of objects' contours. Two recent attempts to adapt video-tracking concepts to provide a precise multi-object segmentation [71, 72] fail completely when objects cross each other, as in the *Museum*, *Granados-S3*, and *Loulous* sequences.

In the rest of this section, we describe a novel hybrid technique which combines the benefits of classical video tracking with those of CNN-based semantic segmentation. The structure of our hybrid technique is shown in Fig. 1. CNN-based modules are depicted in green and red; their inner structure is described in Section 3.2.1 and Fig. 2. Modules inspired from

① `https://perso.telecom-paristech.fr/gousseau/video_inpainting/`
② `http://gvv.mpi-inf.mpg.de/projects/vidinp/`

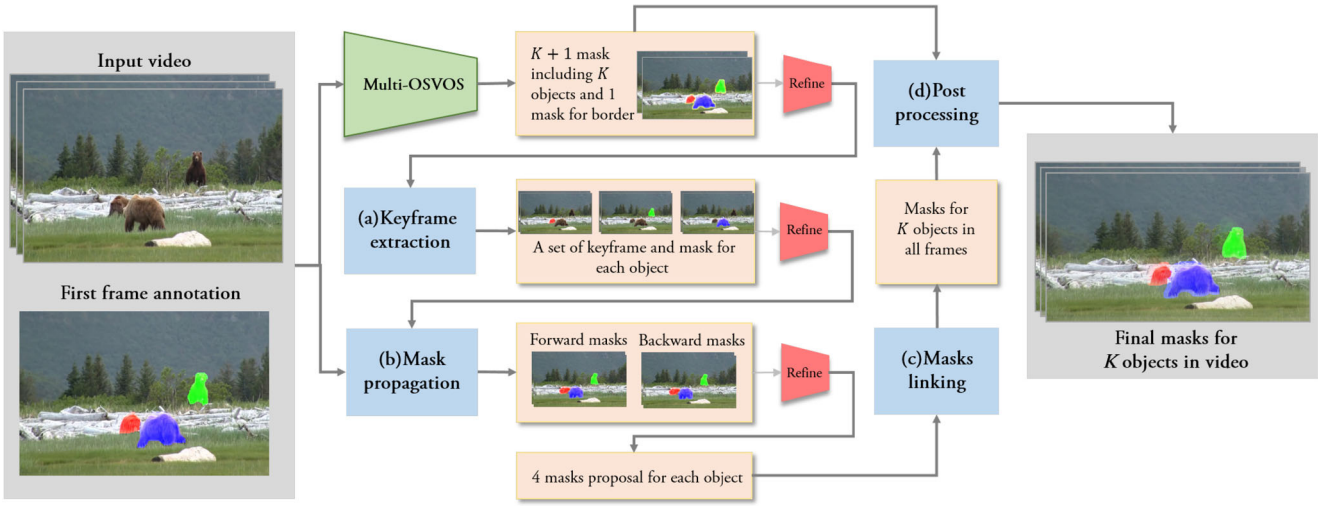


Fig. 1 General pipeline of our object segmentation method. Given the input video and annotations in the first frame, our algorithm alternates two CNN-based semantic segmentation steps (multi-OSVOS network in green, refining network in red) with 4 video-tracking steps (blue blocks): (a) keyframe extraction, (b) mask propagation, (c) mask linking, and (d) post processing. See Section 3.2.

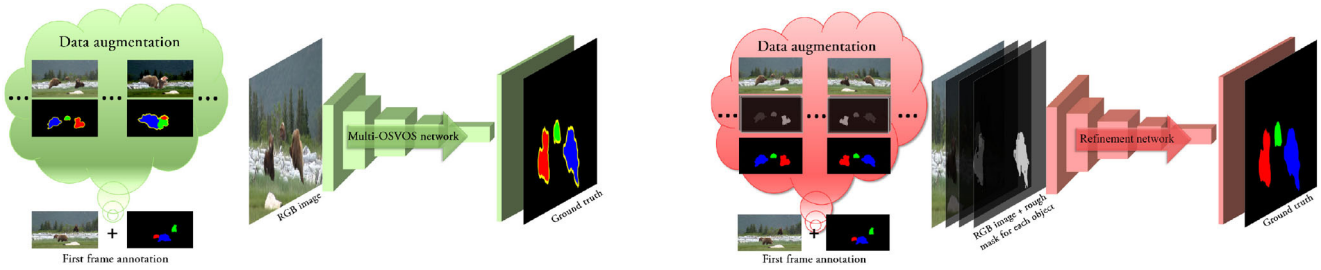


Fig. 2 Two networks used in the general pipeline. Left: multi-OSVOS network, right: refinement network. They serve different purposes: the multi-OSVOS network helps us separating background and objects while the refinement network is used to fine-tune a rough input mask.

video-tracking concepts are depicted in blue and are detailed in Section 3.2.2.

Note that the central part of Fig. 1 operates on a frame-by-frame basis. Each segmentation proposal from the *multi-OSVOS network* (green), or from the *mask propagation* module (blue) is improved by the *refinement network* (red). On the right of the figure, the *mask linking* module (blue) builds a graph that links all segmentation proposals from previous steps, and makes a global decision on the optimal segmentation for each of the K objects being tracked. Finally the *keyframe extraction* module is required to set sensible temporal limits to the *mask propagation* iterations, while the final *post-processing* module further refines the result with the objective of maximizing recall, which is much more important than precision in the case of video inpainting. All these modules are explained in more detail in the following sections.

3.2.1 Semantic segmentation networks

Our system uses two different semantic segmentation networks: a *multi-OSVOS* network and a *refinement* network. Both operate on a frame-by-frame basis.

Our implementation of *multi-OSVOS* computes $K + 1$ masks for each frame: K masks for the K objects of interest and a novel additional mask covering the objects' boundaries. We call this latter mask a *smart dilation* layer; it is the key to guaranteeing that segmentation does not miss any part of the objects, which is especially difficult in the presence of motion blur.

While the *multi-OSVOS network* provides a first prediction, the *refinement network* takes mask predictions as additional guidance input and improves those predictions based on image content, similarly to Ref. [9].

Training these networks is a challenging task, because the only labeled example we can rely

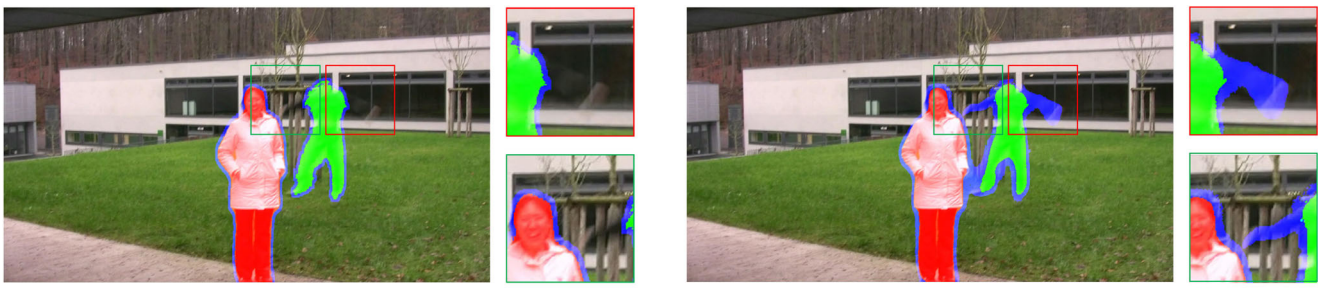
on (for supervised training) is the first annotated frame and the corresponding K masks. The next paragraphs focus on our networks' architectures and on semi-supervised training techniques that we use to circumvent the training difficulty.

Multi-OSVOS network. The training technique of our semantic segmentation networks is mainly inspired from the OSVOS network [8], a breakthrough which achieved the best performance in the DAVIS-2016 challenge [7]. The OSVOS network uses a transfer learning technique for image segmentation: the network is first pre-trained on a large database of labeled images. After training, this so-called *parent* network can roughly separate all foreground objects from the background. Next, the parent network is fine-tuned using the first frame annotation (annotation mask and image) in order to improve the segmentation of a particular object of interest. OSVOS has proven to be a very fast and accurate semi-supervised method to obtain background–foreground separation. Our multi-OSVOS network uses a similar transfer learning technique, yet with several important differences:

- Our network can identify different objects separately (instead of simply foreground and background) and provides a smart dilation mask, i.e., a smart border which covers the interfaces between segmented objects and the background, significantly reducing the number of false negative pixels. The ground truth for this smart dilation mask is defined in the fine-tuning step by a 7-pixel wide dilation of the union of all object masks.
- Unlike OSVOS, which uses a fully convolutional network (FCN) [73], our network uses the Deeplab v2 [74] architecture as the parent model since it outperforms FCN on some common datasets such as PASCAL VOC 2012 [75].
- In the fine-tuning training step we adopt a data augmentation technique in the spirit of Lucid Tracker [21]: we remove all objects from the first frame using Newson et al.'s image inpainting algorithm [76], then the removed objects undergo random geometric deformations (affine and thin plate deformations), and eventually are Poisson blended [77] over the reconstructed background. This is a sensible way of generating large amounts of labeled training data with an appearance similar to that which the network might observe in the following frames.

The smart dilation mask is of particular importance to ensure that segmentation masks do not miss any part of the object, which is typically difficult in the presence of motion blur. A typical example can be seen in Fig. 3 where some parts of the man's hands and legs cannot be captured by simply dilating the output mask, because motion blur leads to partially transparent zones which are not recognized by the network as part of the man's body. With the smart dilation mask, the missing parts are properly captured, and there are no left over pixels.

Refinement network. The multi-OSVOS network can separate objects and background precisely, but it relies exclusively on how they appear in the annotated frame without consideration of their positions, shapes, or motion cues across frames. Therefore, when objects have similar appearance, multi-OSVOS fails to separate individual object instances. In order to take such cues into account, we propagate and compare the prediction of multi-OSVOS across frames using video tracking techniques (see Section 3.2.2), and then double-check and improve the result after each tracking step using the refinement network described below.



(a) (b)

Fig. 3 Advantages of using the smart dilation mask, a smart border layer in the output map of our multi-OSVOS network. (a) Border obtained by simply dilating the output map of the network: some parts of the objects are not covered. (b) Border layer learned by the network: the transition region is covered.

The refinement network has the same architecture as the multi-OSVOS network, except that (i) it takes an additional input, namely mask predictions for the K foreground objects from another method, and (ii) it does not produce as an output the $(K+1)$-th smart dilation mask that does not require any further improvement for our purposes.

Training is performed in exactly the same way as for multi-OSVOS, except that the training set has to be augmented with inaccurate input mask predictions. These should not be exactly the same as the output masks; otherwise, the network would learn to perform a trivial operation ignoring the RGB information. Such inaccurate input mask predictions are created by applying relevant random degradations to ground truth masks, e.g., small translations, and affine and thin-plate spline deformations, followed by a coarsening step (morphological contour smoothing and dilation) to remove details of the object contour; finally, some random tiny square blocks are added to simulate common errors in the output of multi-OSVOS. The ground truth output masks in the training dataset are also dilated by a structuring element of size 7×7 pixels in order to have a safety margin which ensures that the mask does not miss any part of the object.

3.2.2 Multiple object tracking

As a complement to CNN-based segmentation, we use more classical video tracking techniques in order to take global motion and position information into account. The simplest ingredient of our object tracking subsystem is a motion-based *mask propagation* technique that uses a patch-based similarity measure to propagate a known mask to the consecutive frames. It corresponds to block (b) in Fig. 1 and is described in more detail below. This simple scheme alone can provide results similar to other object tracking methods such as SeamSeg [71] or ObjectFlow [72]. In particular it is able to distinguish between different instances of similar objects, based on motion and position. However it loses track of the objects when they cross each other, and it accumulates the errors. To prevent this from happening we complement the mask propagation module with five coherence reinforcement steps:

Semantic segmentation. The refinement network (see Section 3.2.1) is applied to the output of each mask propagation step in order to avoid errors accumulating from one frame to the next.

Keyframe extraction. Mask propagation is effective only when it propagates from frames where object masks are accurate (especially when objects do not cross each other). Frames where this is detected to be true are labeled as *keyframes*, and mask propagation is performed only between pairs of successive keyframes.

Mask linking. When the mask propagation step is unsure about which decision to make, it provides not one, but several mask candidates for each object. A graph-based technique allows all these mask candidates to be linked together. In this way, the decision on which mask candidate is best for a given object on a given frame is made based on global motion and appearance information.

Post-processing. After mask linking, a series of post-processing steps are performed using the original multi-OSVOS result to expand labelling to unlabelled regions.

Interactive correction. In some situations where errors appear, the user can manually correct them on one frame and this correction is propagated to the remaining frames by the propagation module.

The following paragraphs describe in detail the inner workings of the four main modules of our multiple object tracking subsystem: (i) keyframe extraction, (ii) mask propagation, (iii) mask linking, and (iv) post-processing.

Keyframe extraction. A frame t is a keyframe for an object $i \in \{1, \ldots, K\}$ if the mask of this particular object is known or can be computed with high accuracy. All frames in which the object masks were manually provided by the user are considered keyframes. This is usually the first frame or a very few representative frames.

The remaining frames are considered keyframes for a particular object when the object is clearly isolated from other objects and the mask for this object can be computed easily. To quantify this criterion, we rely on the multi-OSVOS network which returns $K+1$ masks O_i for each frame t and $i \in \{1, \ldots, K+1\}$. This allows us to compute the global foreground mask $F = \bigcup_{i=1}^{K+1} O_i$. To verify whether this frame is a keyframe for object $i \in \{1, \ldots, K\}$ we proceed as follows:

1. Compute the connected components of O_i. Let O_i' represent the largest connected component.

2. Compute the set of connected components of the global foreground mask F and call it $\mathcal{F}$.

3. For each connected component $O' \in \mathcal{F}$ compute the overlap ratio with the current object $r_i(O') = |O'_i \cap O'|/|O'|$. If $r_i(O') > 80\%$ and both O'_i and O' are isolated from the remaining objects[①] then this is a keyframe for object i.

Mask propagation. Masks are propagated forwards and backwards between keyframes to ensure temporal coherence. More specifically, forward propagation proceeds as follows: given the mask M_t at frame t, the propagated mask M_{t+1} is constructed with the help of a patch-based nearest neighbor shift map ϕ_t from frame $t+1$ to frame t, defined as

$$\phi_t(p) := \operatorname*{argmin}_{\delta} \underbrace{\sum_{q \in N_p} \| u_{t+1}(q) - u_t(q+\delta) \|^2}_{d^2(D_{t+1}(p), D_t(p+\delta))}$$

i.e., it is the shift δ that minimizes the squared Euclidean distance between the patch centered at pixel p in frame $t+1$ and the patch around $p+\delta$ at frame t. In this expression, N_p denotes a square neighborhood of given size centered at p, and $D_t(p)$ is the associated patch in frame t, i.e., $D_t(p) = u_t(N_p)$ with u_t the RGB image corresponding to frame t. The ℓ^2-metric between patches is denoted d. To improve robustness and speed, this shift map is often computed using an approximate nearest neighbor searching algorithm such as coherency sensitive hashing (CSH) [78], or FeatureMatch [79]. To capture the connectivity of patches across frames in the video, two additional terms are used in Ref. [71] for space and time consistency: the first term penalizes the absolute shift and the latter penalizes neighbourhood incoherence to ensure adjacent patches flow coherently. Moreover, to reduce the patch space dimension and to speed up the search, all patches are represented with lower dimension features, e.g., the main components in Walsh–Hadamard space; see Ref. [71] for more details. We use this model to calculate our shift map.

Once the shift map has been computed, we propagate the mask as follows: let $u_t(p)$ be the RGB value of pixel p in frame t. Then the similarity between a patch $D_{t+1}(p)$ in frame $t+1$ and its nearest neighbour $D_t(p+\phi(p))$ in frame t is measured as

$$s_p = \exp\left(-d^2(D_{t+1}(p), D_t(p+\phi_t(p)))\right)$$

[①] i.e., if $O'_i \cap O'_j = O' \cap O'_j = \emptyset$ for all $j \in \{1, \ldots, K\}$ such that $j \neq i$.

Using this similarity measure the mask M_{t+1} is propagated from M_t using the following rule:

$$\tilde{M}_{t+1}(p) = \begin{cases} 1, & \sum_{q \in N_p} s_q M_t(q) > \frac{1}{2} \sum_{q \in N_p} s_q \\ 0, & \text{otherwise} \end{cases}$$

The final propagated mask M_{t+1} is obtained by a series of morphological operations including opening and hole filling on $\tilde{M}_{t+1}$ followed by the refinement network to correct certain errors. Then M_{t+1} is iteratively propagated to the next frame $t+2$ using the same procedure until we reach the next keyframe.

Although this mask propagation approach is useful, several artifacts may occur when objects cross each other: the propagation algorithm may lose track of an occluded object or it could mistake one object for another. To avoid such errors, mask propagation is performed in both forwards and backwards directions between keyframes. This gives for each object two candidate masks at each frame t: $M_t^1 = M_t^{\mathrm{FW}}$, i.e., the one that has been forward-propagated from a previous keyframe $t' < t$ and $M_t^2 = M_t^{\mathrm{BW}}$, i.e., the one that has been backward-propagated from an upcoming keyframe $t' > t$. In order to circumvent both lost and mistaken objects we consider for each object two additional candidate masks:

$$M_t^3 = M_t^{\mathrm{FW}} \cap M_t^{\mathrm{BW}} \quad \text{and} \quad M_t^4 = M_t^{\mathrm{FW}} \cup M_i^{\mathrm{BW}}$$

The decision between these four mask candidates for each frame and each object is deferred to the next step, which makes that decision based on global optimization.

Mask linking. After backward and forward propagation, each object has 4 mask proposals (except for keyframes where it has a single mask proposal). In order to decide which mask to pick for each object in each frame, we use a graph-based data association technique (GMMCP) [80] that is specially well-suited to video tracking problems. This technique not only allows selection among the 4 candidates for a given object on a given frame, it is also capable of correcting erroneous object-mask assignments on a given frame, based on global similarity computations between mask proposals along the whole sequence. The underlying generalized maximum multi-cliques problem is clearly NP-hard, but the problem itself is of sufficiently small size to be handled effectively by a fast binary-integer program as in Ref. [80].

Formally, we define a complete undirected graph $G = (V, E)$ where each vertex in V corresponds to a mask proposal. Vertices in the same frame are

grouped together to form a cluster. E is the set of edges connecting any two different vertices. Each edge $e \in E$ is weighted by a score measuring the similarity between the two masks it connects, as detailed in the next paragraph. All vertices in different clusters are connected together. The objective is to pick a set of K cliques[①] that maximize the total similarity score, with the restriction that each clique contains exactly one vertex from each cluster. Each selected clique represents the most coherent tracking of an object across all frames.

Region similarity for mask linking. In our graph-based technique, a score needs to be specified to measure the similarity between the two masks, and the associated image data. This similarity must be robust to illumination changes, shape deformation, and occlusion. Many previous approaches in multiple object tracking [80, 81] have focused on global information of the appearance model, typically the global histogram, or motion information (given by optical flow or a simple constant velocity assumption). However, when dealing with large displacements and with an unstable camera, the constant velocity assumption is invalid and optical flow estimation is hard to apply. Furthermore, using only global information is insufficient, since our object regions already have similar global appearance. To overcome this challenge, we define our similarity score as a combination of global and local features. More precisely, each region R is described by the corresponding mask M, its global HSV histograms H, a set P of SURF keypoints [82], and a set E of vectors which connect each keypoint with the centroid of the mask. Each region is determined by four elements:

$R := (M, H, P, E)$,

$P := \{p_1, p_2, \ldots, p_N \,|\, p_i \in M\}$, where p_i is the ith keypoint,

$E := \{\vec{e_1}, \vec{e_2}, \ldots, \vec{e_N} \,|\, \vec{e_i} = p_i - C\}$, where C is the barycenter of M.

Then the similarity between two regions is defined as:

$$S(R_1, R_2) = S_H(R_1, R_2) + \alpha S_P(R_1, R_2)$$

In this expression, $S_H(R_1, R_2) = \exp(-d_c(H_1, H_2))$ where d_c is the cosine distance between two HSV histograms which encode global color information, S_P is the local similarity computed based on keypoint matching, and α is a balance coefficient to specify the contribution of each component. S_P is computed by

$$S_P(R_1, R_2) = \sum_{p_i \in P_1} \sum_{p_j \in P_2} \gamma_{ij}.w_{ij}$$

where γ_{ij} is the indicator function set to 1 if two keypoints p_i and p_j match, and 0 otherwise. This function is weighted by w_{ij} based on the position of the matching keypoints with respect to the centroid of the region:

$$w_{ij} = \exp\left(-d(\vec{e_i}, \vec{e_j})/(2\sigma)\right)$$

where d_c is the cosine distance between two vectors and σ is a constant.

Post-processing. At this time, we already have K masks for K objects for all frames of the video. Now we perform a post-processing step to ensure that our final mask covers all details of the objects. This is very important in video object removal since any missing detail can cause perceptually annoying artifacts in the object removal result. This post-processing includes two main steps.

The first step is to give a label for each region in the global foreground mask $F_t = \bigcup_{i=1}^{K} O_t^i$ (the union of all object masks produced by multi-OSVOS for frame t) which does not yet have any label. To this end, we proceed as follows. First, we compute the connected components C of all masks O_t^i and try to assign a label to all pixels in each connected component. To do so, we consider the masks M_t^j that were obtained for the same frame t (and possibly another object class j by the mask linking method). A connected component is considered as isolated if $C \cap M_t^j$ is empty for all j. For non-isolated components, a label is assigned by a voting scheme based on the ratio $r_j(C) = \frac{|C \cap M_t^j|}{|C|}$, i.e., the assigned label for region C is $\hat{j} = \text{argmax}_j\, r_j(C)$, the one with the highest ratio. If $r_j(C) > 80\%$ then region C is also assigned label j regardless of the voting result, which may lead to multiple labels per pixel.

In the second step, we do a series of morphological operations, namely opening and hole filling. Finally we dilate each object mask again with size 9×9, this time allowing overlap between objects.

3.3 Object removal

After using the method in the previous section, all selected objects have been segmented throughout the complete video sequence. From the corresponding

① A clique is a subgraph in which every pair of distinct vertices is connected.

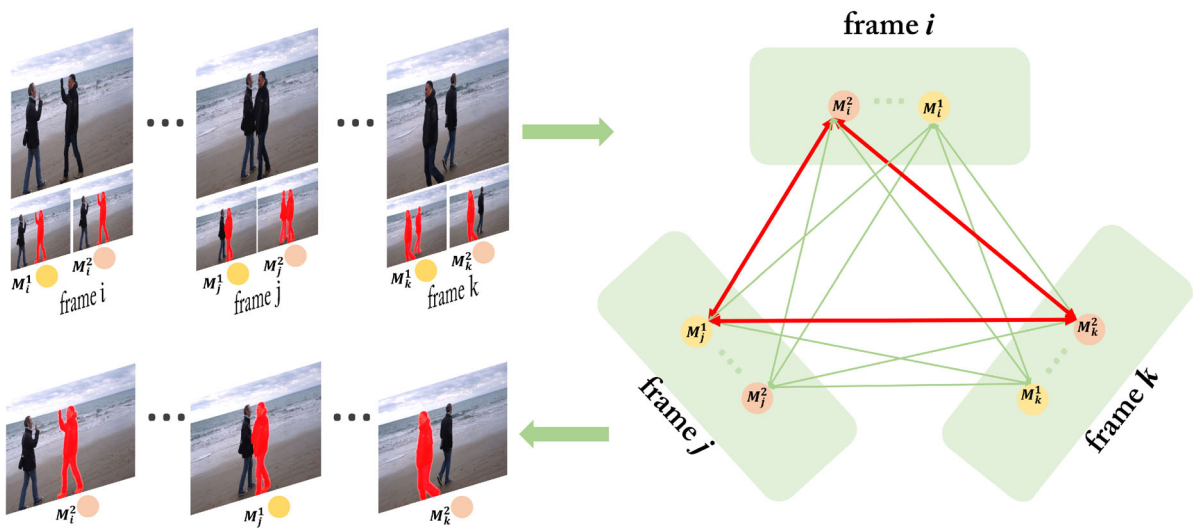


Fig. 4 Mask proposals are linked across frames to form a graph. The goal is then to select a clique from this graph minimizing the overall cost. As a result, a best candidate is picked for each frame to ensure that the same physical object is tracked.

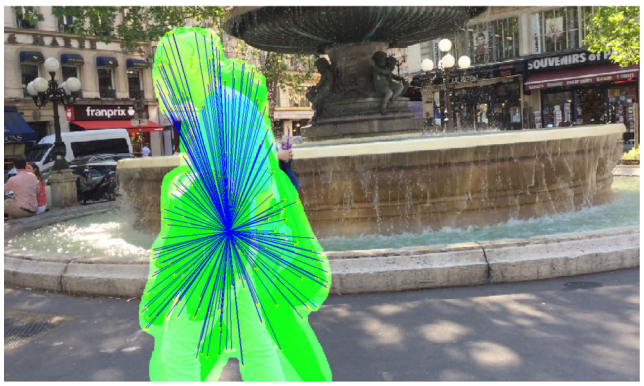


Fig. 5 Region description. Each region is described by a global histogram, a set of SURF keypoints (yellow points), and a set of vectors which connects each keypoint and the centroid of the region.

masks, the user can then decide which objects are to be removed. This last step is performed using video inpainting techniques that we now detail. First, we present a simple inpainting method that is used when the background is static (or can be stabilized) and revealed at some point in the sequence. This first method is fast and relies on the reconstruction of a motion field. Then we present a more involved method for the case where the background is moving, with possibly some complex motion as in the case of dynamic textures.

3.3.1 Static background

We assume for this first inpainting method that the background is visible at least in some frames (for instance because the object to be removed is moving over a large enough distance). We also assume that the background is rigid and that its motion is only due to camera motion. In this case, the best option for performing inpainting is to copy the visible parts of the background into the missing regions, from either past or future frames. For this, the idea is to rely on a simple optical-flow pixel propagation technique. Motion information is used to track the missing pixels and establish a trajectory from the missing region toward the source region.

Overview. Our optical flow-based pixel propagation approach is composed of three main steps, as illustrated in Fig. 6. After stabilizing the video to compensate for camera movement, we use FlowNet 2.0 to estimate forward and backward optical flow fields. These optical flow fields are then inpainted using a classical image inpainting method to fill in the missing information. Next, these inpainted motion fields are concatenated to create a correspondence map between pixels in the inpainting region and known pixels. Lastly, missing pixels are reconstructed by a copy-paste scheme followed by Poisson blending to reduce artifacts.

Motion field reconstruction. A possible approach to optical flow inpainting is smooth interpolation, for instance, in the framework of a variational approach, by ignoring the data term and using only the smoothness term in the missing regions, as proposed in Refs. [56, 57]. However, this approach leads to over-smoothed and unreliable optical flow. Therefore, we choose to reconstruct the optical flow using more sophisticated image inpainting

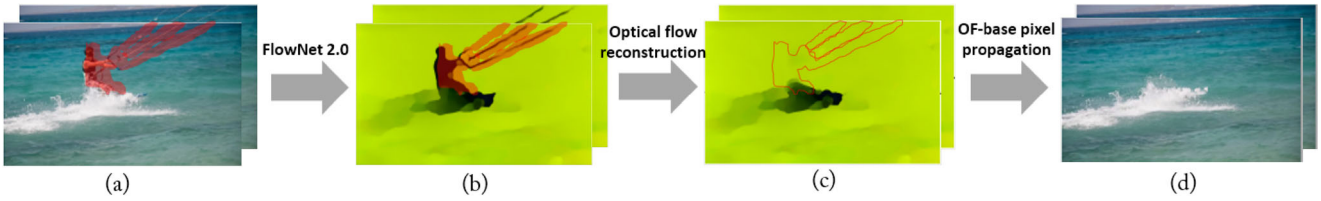


Fig. 6 Global pipeline of the optical flow-based propagation approach for reconstructing a static background. From input video (a), forward/backward optical flow fields are estimated by FlowNet 2.0 (b), and then are inpainted by an image inpainting algorithm (c). From these optical flow fields, pixels from the source region are propagated into the missing region (d).

techniques. More specifically we first compute, outside the missing region, forward and backward optical flow fields between pairs of consecutive frames using the FlowNet approach from Refs. [83]. We then rely on the image inpainting method from Ref. [76] to interpolate these motion fields.

Optical flow-based pixel reconstruction. Once the motion field inside the missing region is filled, it is used to propagate pixel values from the source toward the missing regions. For this to be done, we map each pixel in the missing region to a pixel in the source region. This map is obtained by accumulating the optical flow field from frame to frame (with bilinear interpolation). We compute both forward and backward optical flow, which leads us to two correspondence maps: a forward map and a backward map. From either map, we can reconstruct missing pixels with a simple copy-paste method, using known values outside the missing region.

We perform two passes: first a forward pass using the forward map to reconstruct occlusion, then a backward pass using the backward map. After these two passes, the remaining missing information corresponds to parts that have never been revealed in the video. To reconstruct this information, we first use the image inpainting method from Ref. [76] to complete one keyframe, which is chosen to be the middle frame of the video, and then propagate information from this frame to other frames in the video using forward and backward maps.

Poisson blending. Videos in real life often contain illumination changes, especially if recorded outdoors. This is problematic for our approach that simply copies and pastes pixel values. When illumination of the sources differs from the illumination of the restored frame, visible artifacts across the border of the occlusion may appear. A common way to resolve this is to apply a blending technique, e.g., Poisson blending [77], which fuses a source image and a target image in the gradient domain. However, performing Poisson blending frame-by-frame may affect temporal consistency. To maintain it, we adopt the recent method of Bokov and Vatolin [57] which takes into account information from the previous frame. In this method, a regularizer penalizes discrepancies between the reconstructed colors and corresponding colors in the optical flow-aligned previous frame. More specifically, given the colors of the current and previous inpainted frames $I_t(p)$, $I_{t-1}(p)$, respectively, the refined Poisson-blended image $I(p)$ can be obtained by minimizing the discretized energy functional [57]:

$$\begin{aligned} B(I) = &\sum_{p\in\Omega_t} \|\nabla I(p) - G_t(p)\|^2 \\ &+ \sum_{p\in\partial\Omega_t} w_p^{\mathrm{PB}} \|I(p) - I_t(p)\|^2 \\ &+ \sum_{p\in\Omega_t} (1 - w_p^{\mathrm{PB}}) \|I(p) - I_{t-1}(p + O_t(p))\|^2 \end{aligned}$$

Here, $\partial\Omega_t$ denotes the outer-boundary pixels of the missing region Ω_t, $G_t(p)$ is the target gradient field and $O_t(p)$ is the optical flow at position p between frames $t-1$ and t. The terms w_p^{PB} are defined as

$$w_p^{\mathrm{PB}} = (1 + \sigma^{\mathrm{PB}} ||\nabla I^{\mathrm{PB}}(p) - G_t(p)||^2)^{-1}$$

where I^{PB} is the usual Poisson blended image. They are used to weight the reconstruction results from the previous frame I_{t-1} in the boundary conditions. In this definition, σ^{PB} is a constant controling the strength of temporal-consistency enforcement. These weights allow to better handling of global illumination changes while enforcing temporal stability. This Poisson blending technique is applied at every pixel propagation step to support the copy-paste framework.

3.3.2 Dynamic background

The simple optical flow-based pixel propagation method proposed in Section 3.3.1 can produce plausible results if the video contains only a static

background and simple camera motion. More involved methods are needed to deal with large pixel displacements and complex camera movements. They are typically based on joint estimation of optical flow and color information inside the occlusion: see for instance Refs. [61, 84]. However, when the background is dynamic or contains moving objects, these latter methods often fail to capture oscillatory patterns in the background. In such situations, global patch-based methods are preferred. They rely on minimization of a global energy computed over space–time patches. This idea was first proposed in Ref. [58], later improved in Ref. [59], and recently improved further in Le et al. [62].

We describe briefly the method in Ref. [62]. A prior stabilization process is applied to compensate for instabilities due to camera movement (see below for the improvement proposed in the current work). Then a multiscale coarse-to-fine scheme is used to compute a solution to the inpainting problem. The general structure of this scheme is that, at each scale of a multiscale pyramid, we alternate until convergence (i) computation of an optimal shift map between pixels in the inpainting domain and pixels outside (using a metric between patches which involves image colors, texture features, and optical flow), and (ii) update of image colors inside the inpainting domain (using a weighted average of the values provided by the shift map). A key to the quality of the final result is the coarse initialization of this scheme; it is obtained by progressively filling in the inpainting domain (at the coarsest scale) using patch matching and (mapped) neighbors averaged together with a priority term based on optical flow. The heavy use of optical flow at each scale greatly helps to enforce temporal consistency even in difficult cases such as dynamic backgrounds or complex motions. In particular, the method can reconstruct moving objects even when they interact. The whole method is computationally heavy but the speed is significantly boosted when all steps are parallelized.

We have recently brought several improvements to this method of Ref. [62]:

Video stabilization. In general, patch-based video inpainting techniques require good video stabilization as a pre-processing step to compensate for patch deformations due to camera motions [85, 86]. This video stabilization is usually done by calculating a homography between pairs of consecutive frames using keypoint matching followed by an RANSAC algorithm to remove outliers [87]. However, large moving objects appearing in the video may reduce the performance of such an approach as too many keypoints may be selected on these objects and prevent the homography from being estimated accurately from the background. This problem can be solved by simply neglecting all segmented objects when computing the homography. This is easy to do: since we already have masks of the selected objects, we just have to remove all keypoints which are covered by masks. This is an advantage of our approach in which both segmentation and inpainting are addressed.

Background and foreground inpainting. In addition to stabilization improvement, multiple segmentation masks are also helpful for inpainting separately the background and the foreground. More precisely, we first inpaint the background neglecting all pixels contained in segmented objects. After that, we inpaint in priority order the segmented objects that we want to keep and which are partially occluded. This increases the quality of the reconstruction, both for the background and for the objects. Furthermore, it reduces the risk of blending segmented objects which are partially occluded because segmented objects have separate labels. In particular, it is extremely helpful when several objects overlap.

We finally mention another advantage of our joint tracking/inpainting method: objects are better segmented and thus easier to inpaint, as it is a well-known fact that the inpainting of a missing domain may be of lower quality if the boundary values are unsuitable. In our case, time continuity of segmented objects and the fact of using different labels for different objects have a huge impact on the quality of the inpainting.

4 Results

We first evaluate our results for the segmentation step of the proposed method. We provide quantitative and visual results, and comparisons with state-of-the-art methods. We then provide several visual results for the complete object removal process, again comparing with the most efficient methods. These visual comparisons are given as isolated frames in the paper; it is of course more informative to see

the complete videos in the supplementary material at `https://object-removal.telecom-paristech.fr/`. We consider various datasets: we use sequences from the DAVIS-2016 [7] challenge, and from the MOViC [88] and ObMIC [89] datasets; we also consider classical sequences from Refs. [49] and [59]. Finally, we provide several new challenging sequences containing strong appearance changes, motion blur, objects with similar appearance and at times crossing, as well as complex dynamic textures.

Unless otherwise stated, only the *first* frame is annotated by the user in all experiments. In some examples (e.g., *Camel*) not all objects are visible in the first frame and we use another frame for annotation. In a few examples we annotate more than one frame (e.g., the first and last frame in *Teddy bear-fire and Jumping girl-fire*) in order to illustrate the flexibility of the system in correcting errors.

4.1 Implementation details

For segmentation, we use the Deeplab v2 [74] architecture for the multi-OSVOS and refining networks. We initialize the network using the pre-trained model provided by Ref. [74] and then adapt it to video using the training set of DAVIS-2016 [69] and the *train-val* set in DAVIS-2017 [19] (we exclude the validation set of DAVIS-2016). For the data augmentation procedure, we generate 100 pairs of images and ground truth from the first frame annotation, following the same protocol as in Ref. [21]. For the patch-based mask propagation and mask linking, we evolved from the implementations of Refs. [71] and [80], respectively.

For the video inpainting step, we use the default parameters from our previous work [62]. In particular, the patch size is set to 5, and the number of levels in the multi-scale pyramid is 4.

For a typical sequence with resolution (854×480) and 100 frames, the full computational time is of the order of 45 minutes for segmentation plus 40 minutes for inpainting on an Intel Core i7 CPU with 32 GB of RAM and a GTX 1080 GPU. While this is a limitation of the approach, complete object removal is about one order of magnitude faster than a single completion step from state-of-the-art methods [59, 61]. While interactive editing is out of reach for now, the computational time allows offline post-processing of sequences.

4.2 Object segmentation

For the proposed object removal system, and as explained in detail above, the most crucial point is that the segmentation masks must completely cover the considered objects, including motion and transition blur. Otherwise, unacceptable artifacts remain after the full object removal procedure (see Fig. 13 for an example). In terms of performance evaluation, this means that we favor *recall* over *precision*, as defined below. This also means that the ground truth provided with classical datasets may not be fully adequate to evaluate segmentation in the context of object removal, because they do not include transition zones induced by, e.g., motion blur. For this reason, recent video inpainting methods that make use of these databases to avoid the tedious manual selection of objects usually start from a *dilation* of the ground truth. In our case, a dilation is learned by our architecture (smart dilation) during the segmentation step, as explained above. For these reasons, we compare our method with state-of-the-art object segmentation methods, after various dilations and on the dilated versions of the ground truth. We also provide visual results in our supplementary website at `https://object-removal.telecom-paristech.fr/`.

Evaluation metrics. We briefly recall here the evaluation metrics that we use in this work: some of them are the same as in the DAVIS-2016 challenge [7] and we also add other metrics specialized for our task. The goal is to compare the computed segmentation mask (SM) to the ground truth mask (GT). The *recall* metric is defined as the ratio between the area of the intersection between SM and GT, and the area of GT. The *precision* is the ratio between the area of the intersection and the area of the SM. Finally, the *IoU* (intersection over union), or Jaccard index, is defined as the ratio between intersection and union.

Single object segmentation. We use the DAVIS-2016 [7] validation set and compare our approach to recent semi-supervised state-of-the-art techniques (SeamSeg [71], ObjectFlow [72], MSK [69], OSVOS [8], and onAVOS [20]) using pre-computed segmentation masks provided by the authors. As explained above, we consider a dilated version of the ground truth (using a 15×15 structuring element, as in Refs. [61, 62]). Therefore, we apply a dilation of the same size to the masks from all methods. In our

case, this dilation has both been learned (size 7×7) and applied as a post-processing step (size 9×9). Since the composition of two dilations with such sizes yields a dilation with size 15×15, the comparison is fair.

Table 1 shows a comparison using the three above-mentioned metrics. Our method has the best recall score overall, therefore achieving its objective. The precision score remains very competitive. Besides, our method outperforms OSVOS [8] and MSK [69], those having a similar neural network backbone architecture (VGG16), on all metrics. The precision and *IoU* scores compare favorably with onAVOS [20] which uses a deeper and more advanced network. Table 2 provides a comparison between OSVOS [8] and our approach on two sequences from Ref. [60]. These sequences have been manually segmented by the authors of Ref. [60] for video inpainting purposes. On such extremely conservative segmentation masks (in the sense that they over-detect the object), the advantage of our method is particularly strong.

As a further experiment, we investigate the ability of dilations of various sizes to improve recall without degrading precision too much. For this, we plot precision–recall curves as a function of the structuring element size (ranging from 1 to 30). To include our method on this graph, we start from our original method (highlighted with a green square) and apply to it either erosions with a radius ranging from 1 to 15, or dilation with a radius ranging from 1 to 15. Again this makes sense since our method has learned a dilation whose equivalent radius is 15. Results are displayed in Fig. 8. As can be seen, our method is the best in terms of recall, and recall increases significantly with dilation size. With the sophisticated onAVOS method, on the other hand, recall increases slowly, while precision drops drastically, as dilation size increases. Basically, these experiments show that the performance achieved by our system for the full coverage of a single object (that is, with as few missed pixels as possible) cannot be obtained from state-of-the-art object segmentation methods by using simple dilation techniques.

Multiple object segmentation. Next, we perform the same experiments for datasets containing videos with multiple objects. Since the *test* ground truth was not yet available (at the time of writing) for the DAVIS-2017 dataset and since our network was trained on the *train-val* set of this dataset, we consider two other datasets: MOViCs [88] and ObMIC [89]. They include multiple objects, but only have one label per sequence. To evaluate multiple object situations, we only kept sequences containing more than one object, and then manually re-annotated the ground truth giving different labels for different instances. Observe that these datasets contain several major difficulties such as large camera displacements, motion blur, similar appearances, and crossing objects. Results are summarized in Table 3. Roughly the same conclusions can be drawn as in

Table 1 Quantitative comparison of our object segmentation method to other state-of-the-art methods, on the single object DAVIS-2016 [7] validation set. The main objective when performing object removal is to achieve high recall scores

	Metric		
	Recall (%)	Precision (%)	*IoU* (%)
SeamSeg	59.31	73.08	50.20
ObjectFlow	70.63	90.97	67.78
MSK	82.83	95.00	79.94
OSVOS	86.78	92.38	80.58
onAVOS	87.64	**96.67**	**85.17**
Ours	**89.63**	94.31	84.70

Table 2 Quantitative comparison of our object segmentation method and the OSVOS segmentation method [8], on two sequences manually segmented for inpainting purposes [60]

	Metric		
	Recall (%)	Precision (%)	*IoU* (%)
Granados-S1			
OSVOS	62.04	59.17	52.15
Ours	80.12	86.31	67.53
Granados-S3			
OSVOS	74.42	87.00	63.02
Ours	80.12	86.31	67.53

Table 3 Quantitative comparison of our object segmentation method and other state-of-the-art methods, on two multiple objects datasets (MOVICs [88] and ObMIC [89])

	Metric		
	Recall (%)	Precision (%)	*IoU* (%)
MOViCs			
SeamSeg	78.63	74.06	65.96
ObjectFlow	59.50	77.01	52.33
OSVOS	85.48	83.87	76.63
Ours	**89.28**	**87.09**	**81.58**
ObMIC			
SeamSeg	91.00	80.30	75.33
ObjectFlow	53.14	83.00	43.64
OSVOS	85.89	84.08	74.55
Ours	**94.42**	**88.48**	**83.81**

the single object case, namely the superiority of our method in term of recall, without much sacrifice of precision .

Some qualitative results of our video segmentation technique are shown in Fig. 7. In the first two rows, we show some frames corresponding to the single object case, on the DAVIS-2016 dataset [7]. The last three rows show multiple object segmentation results on MOViCs [88], ObMIC [89], and Granados's sequences [49] respectively. On these examples, our approach yields full object coverage, even with complex motion and motion blur. This is particularly noticeable on the sequences *Kite-surf* and *Paragliding-launch*. In the multiple object cases, the examples illustrate the capacity of our method to deal with complex occlusions. This cannot be achieved with mask tracking methods such as objectFlow [72] or SeamSeg [71]. The OSVOS method [8] leads to some confusion of objects, probably because temporal continuity is not taken into account by this approach.

4.3 Object removal

Next, we evaluate the complete object removal pipeline. We consider both inpainting versions that we have introduced. We use the simple, optical flow-based method introduced in Section 3.3.1 for sequences having static backgrounds. We refer to this fast method as the *static version*. We use the more complex method derived from Ref. [62] and detailed in Section 3.3.2 for more involved sequences, exhibiting challenging situations such as dynamic backgrounds, camera instability, complex motions, and crossing objects. We refer to this second slower version as the *dynamic version*.

In Fig. 9, we display examples of both single and multiple object removal, through several representative frames. The video results can be fully viewed in the supplementary website. The first sequence *Blackswan* (DAVIS-2016) shows that our method (dynamic version) can plausibly reproduce dynamic textures. In the second sequence *Cows* (DAVIS-2016), the method yields good results, with a stable background and continuity of the geometrical structures, despite a large occlusion implying that some regions are covered throughout the sequence. We then turn to the case of multiple object removal. In the sequence *Camel* (DAVIS-2017), we show the removal of one static object, a challenging case since the background information is missing in places. In

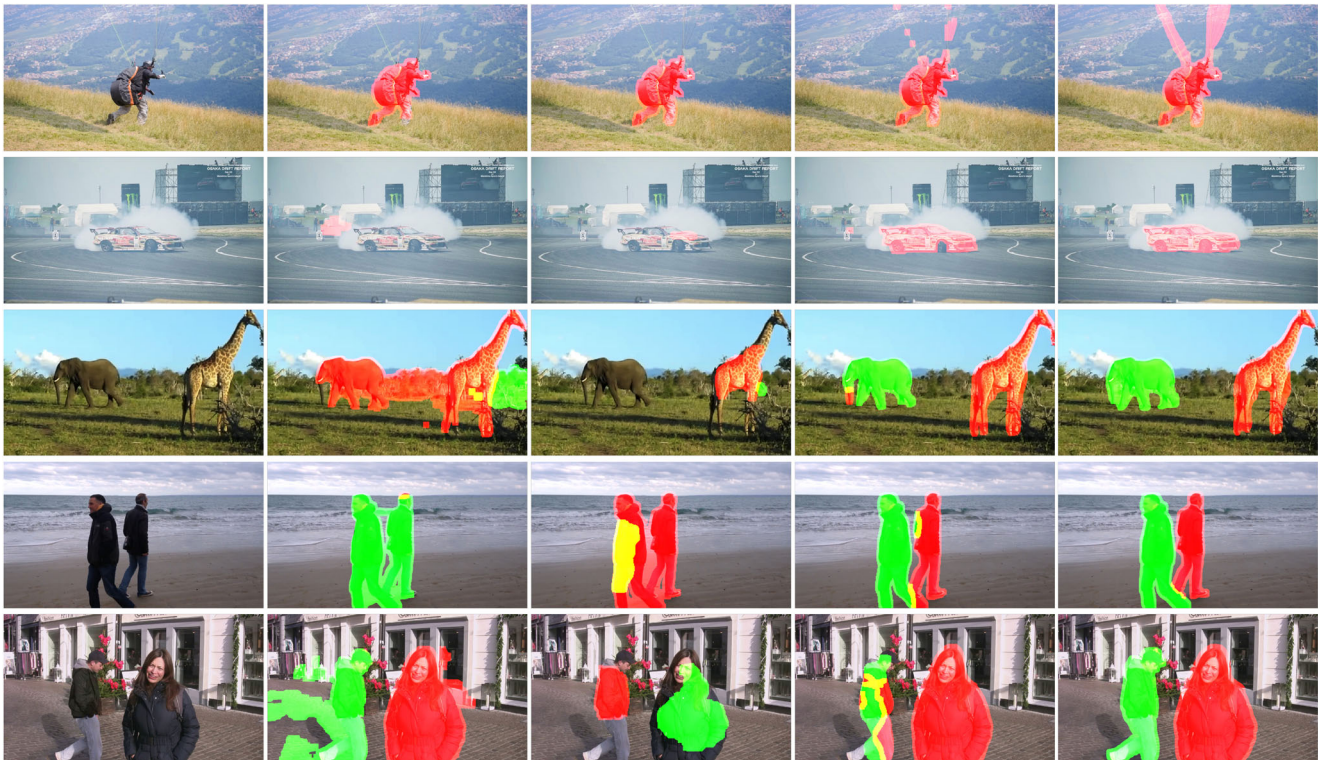


Fig. 7 Visual comparison of different segmentation approaches. Left to right: original, SeamSeg [71], ObjectFlow [72], OSVOS [8], ours.

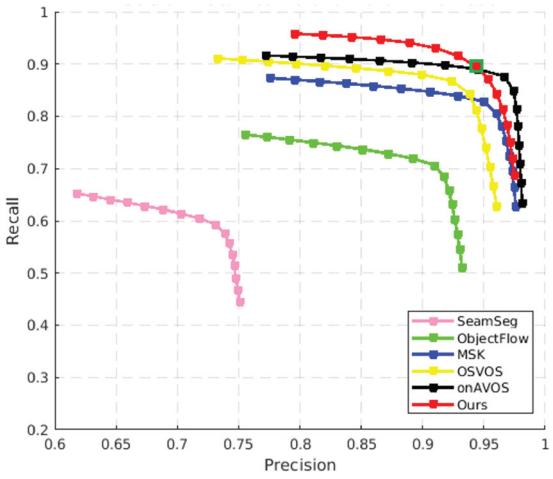


Fig. 8 Precision–recall curves for different methods with different dilation sizes.

this example, the direct use of the inpainting method from Ref. [62] results in some undesired artifacts when the second camel enters the occlusion. By using multiple object segmentation masks to separate background and foreground, we can create a much more stable background. The last two examples are from an original video. This sequence again highlights that our method can deal with dynamic textures and hand-held cameras.

Comparison with state-of-the-art inpainting methods. In these experiments, we compare our results with two state-of-the-art video inpainting methods [59, 61].

First, we provide a visual comparison between our optical flow-based pixel propagation method (the

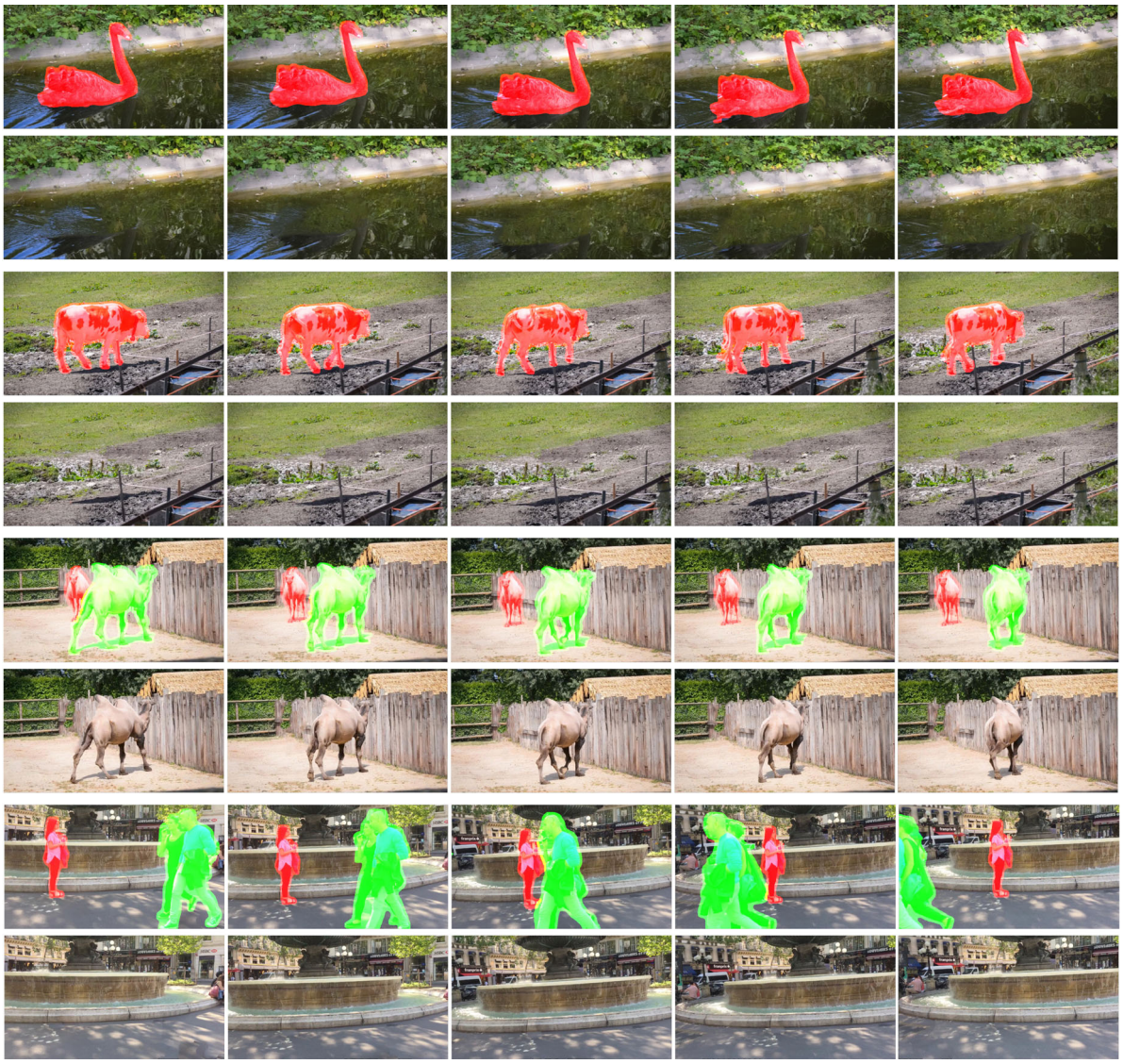


Fig. 9 Results from our object removal system.

static approach) with the method of Huang et al. [61] using a video with a static background. Figure 10 shows some representative frames of the sequence *Horse-jump-high*. In this sequence, we achieve a comparable result using our simple optical flow-based pixel propagation approach. Our advantage is the considerable reduction of computational time. With an unoptimized version of the code, our method takes approximately 30 minutes to finish while the method in Ref. [61] takes about 3 hours to complete this sequence.

Next, we qualitatively compare our method with Ref. [61] when reconstructing dynamic backgrounds. We use the code released by the author on several sequences using the default parameters. In general, Huang et al. [61] fail to generate convincing dynamic textures. This can be explained by the fact that their algorithm relies on dense flow fields to guide completion, and these fields are often unreliable for dynamic texture. Moreover, they fill the hole by sampling only 2D patches from the source regions and therefore the periodic repetition of the background is not captured. Our method, on the other hand, fills the missing dynamic textures in a plausible way. Figure 11 shows representative frames of the reconstructed sequence *Teddy-bear*, which is recorded indoors. This sequence is especially challenging because of the presence of both dynamic and static textures, as well as illumination changes. Our method yields a convincing reconstruction of the fire, unlike the one in Ref. [61]. The complete video can be seen in the supplemental material website.

We also compare our results with the video inpainting technique from Ref. [59]. Figure 12 shows some representative frames of the sequence *Park-complex*, which is taken from Ref. [60] and modified to focus on the moment when objects occlude each other. In this example, the method of Ref. [59] cannot reconstruct the moving man on the right which is occluded by the man on the left. This is

Fig. 10 Qualitative comparison with Huang et al.'s method [61]. Top to bottom: our segmentation mask, result from Ref. [61] using a manually segmented mask, our inpainting results using our mask.

Fig. 11 Qualitative comparison with Huang et al.'s method [61] on video with a dynamic background. Left to right: our segmentation mask, result from Ref. [61], our inpainting result performed on our mask.

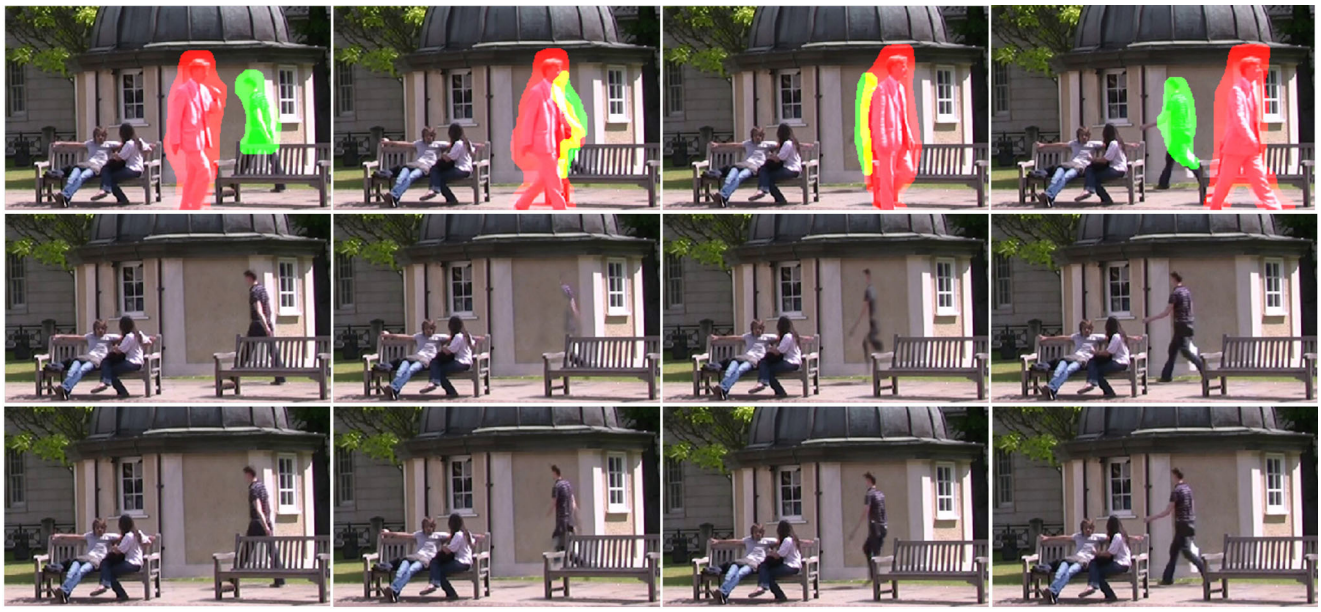


Fig. 12 Qualitative comparison with Newson et al.'s method [59]. Top: our segmentation masks; red and green masks denote different objects, and yellow shows the overlap region between two objects. Middle: results from Ref. [59] performed on our segmentation masks. Bottom: our inpainting results performed on the same masks.

because the background behind this man changes over time (from tree to wall). Since Newson et al.'s method [59] treats the background and the foreground similarly, the algorithm can not reconstruct the situation "man in front of the wall" because it has never seen this situation before. Our method, by making use of the optical flow and thanks to the object segmentation map, can reconstruct the "man" and the "wall" independently, yielding a plausible reconstruction.

Impact of segmentation masks on inpainting performances. In these experiments, we highlight the advantages of using the segmentation masks of multiple objects to improve the video inpainting results.

First, we emphasize the need for masks which fully cover the objects to be removed. Figure 13(top) demonstrates the situation in which some object details (the waving hand in this case) are not covered by the mask (here using the state-of-the-art OSVOS method) [8]. This situation leads to a very unpleasant artifact when video inpainting is performed. Thanks to the smart dilation, introduced in the previous sections, our segmentation mask fully covers the object to be removed, yielding a more plausible video after inpainting.

Object segmentation masks can also be helpful for the video stabilization step. Indeed, in the case of large foregrounds, these can have a strong effect on the stabilization procedure, yielding a bad stabilization of the background, which in turn yields bad inpainting results. In contrast, if stabilization is applied only to the background, the final object removal results are much better. This situation is illustrated in the supplementary material.

To further investigate the advantage of using multiple segmentation masks to separate background and foreground in the video completion algorithm, we compare our method with direct application of the inpainting method from Ref. [62], without separating objects and background. Representative frames ofboth approaches are shown in Fig. 14. Clearly, the method in Ref. [62] produces artifacts when the moving objects (the two characters) overlap the occlusion, due to patches from these moving objects being propagated within the occlusion in the nearest neighbor search step. Our method, on the other hand, does not suffer from this problem because we reconstruct background and moving objects separately. This way, the background is more stable, and the moving objects are well reconstructed.

5 Conclusions, limitations, and discussion

In this paper, we have provided a full system for performing object removal from video. The input of

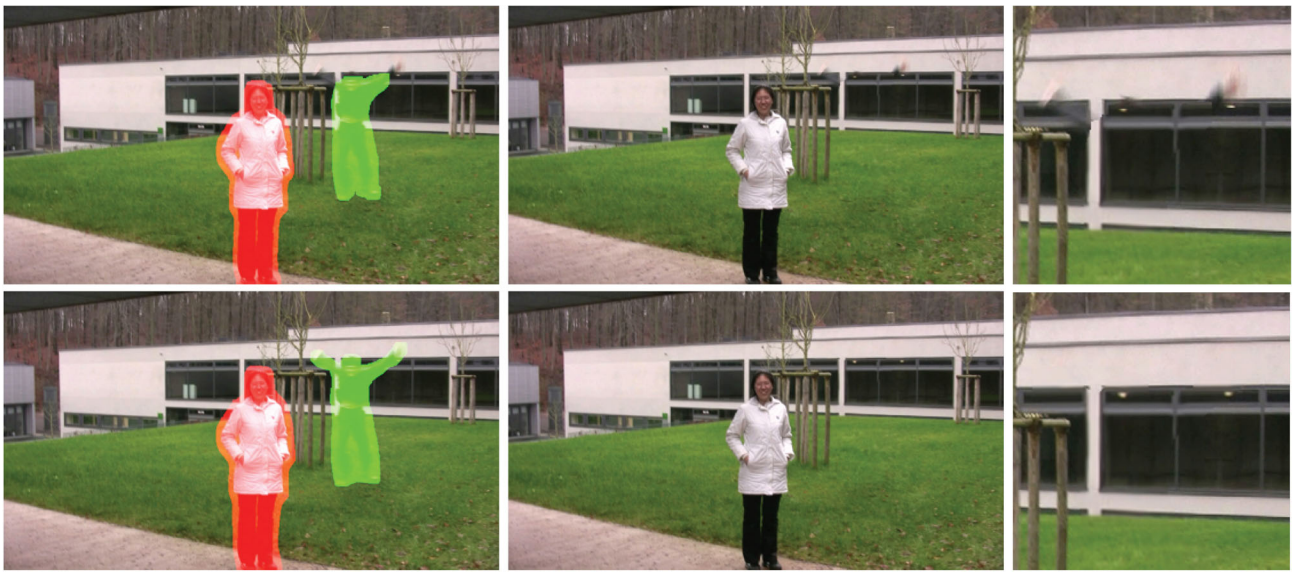


Fig. 13 Results of object removal using masks computed by OSVOS (top) and our method (bottom). Left to right: segmentation mask, resulting object removal in one frame, zooms. When the segmentation masks do not fully cover the object (OSVOS), the resulting video contains visible artifacts (the hand of the man remains after object removal).

Fig. 14 Advantage of using segmentation masks to separate background and foreground. Left: without separation, the result has many artifacts. Right: the background and foreground are well reconstructed when reconstructed independently.

the system comprises a few strokes provided by the user to indicate the objects to be removed. To the best of our knowledge, this is the first system of this kind, even though the Adobe company has recently announced it is developing such a tool, under the name *Cloak*. The approach can deal with multiple, possibly crossing objects, and can reproduce complex motions and dynamic textures.

Although our method achieves good visual results on different datasets, it still suffers from a few limitations. First, parts of objects to be edited may be ignored by the segmentation masks. In such cases, as already emphasized, the inpainting step of the algorithm will amplify the remaining parts, creating strong artifacts. This is an intrinsic problem of the semi-supervised object removal approach and room remains for further improvement. Further, the system is still relatively slow, and in any case far from real time. Accelerating the system could allow for interactive scenarios where the user can gradually correct the segmentation-inpainting loop.

The segmentation of shadows is still not flawlessly performed by our system, especially when the shadows lack contrast. It is a desirable property of the system to be able to deal with such cases. This problem can be seen in several examples provided in the supplementary material.

Concerning the inpainting module, the user has to currently choose between the fast motion-based version (which works better for static backgrounds) and the slower patch-based version which is required in the presence of complex dynamic backgrounds. An

integrated method that reunites the advantages of both would be preferable. Huang et al.'s method [61] makes a nice attempt in this direction, but its use of 2D patches is insufficient to correctly inpaint complex dynamic textures, which are more plausibly inpainted by our 3D patch-based method.

Another limitation occurs in some cases where the background is not revealed, specifically when semantic information should be used. Such difficult cases are gradually being solved for single images by using CNN-based inpainting schemes [64]. While the training step of such methods is still out of reach for videos as of today, developing an object removal scheme fully relying on neural networks is an exciting research direction.

Acknowledgements

We gratefully acknowledge the support of NVIDIA who donated a Titan Xp GPU used for this research. This work was funded by the French Research Agency (ANR) under Grant No. ANR-14-CE27-001 (MIRIAM).

References

[1] Bai, X.; Wang, J.; Simons, D.; Sapiro, G. Video SnapCut: Robust video object cutout using localized classifiers. *ACM Transactions on Graphics* Vol. 28, No. 3, Article No. 70, 2009.

[2] Le, T. T.; Almansa, A.; Gousseau, Y.; Masnou, S. Removing objects from videos with a few strokes. In: Proceedings of the SIGGRAPH Asia Technical Briefs, Article No. 22, 2018.

[3] Wang, S.; Lu, H.; Yang, F.; Yang, M.-H. Superpixel tracking. In: Proceedings of the IEEE International Conference on Computer Vision, 1323–1330, 2011.

[4] Levinkov, E.; Tompkin, J.; Bonneel, N.; Kirchhoff S.; Andres, B.; Pfister, H. Interactive multicut video segmentation. In: Proceedings of the 24th Pacific Conference on Computer Graphics and Applications: Short Papers, 33–38, 2016.

[5] Marki, N.; Perazzi, F.; Wang, O.; Sorkine-Hornung, A. Bilateral space video segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 743–751, 2016.

[6] Nagaraja, N. S.; Schmidt, F. R.; Brox, T. Video segmentation with just a few strokes. In: Proceedings of the IEEE International Conference on Computer Vision, 3235–3243, 2015.

[7] Perazzi, F.; Pont-Tuset, J.; McWilliams, B.; van Gool, L.; Gross, M.; Sorkine-Hornung, A. A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 724–732, 2016.

[8] Caelles, S.; Maninis, K. K.; Pont-Tuset, J.; Leal-Taixé, L.; Cremers, D.; van Gool, L. One-shot video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5320–5329, 2017.

[9] Perazzi, F.; Khoreva, A.; Benenson, R.; Schiele, B.; Sorkine-Hornung, A. Learning video object segmentation from static images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3491–3500, 2017.

[10] Caelles, S.; Chen, Y.; Pont-Tuset, J.; van Gool, L. Semantically-guided video object segmentation. *arXiv preprint* arXiv:1704.01926, 2017.

[11] Dai, J. F.; He, K. M.; Sun, J. Instance-aware semantic segmentation via multi-task network cascades. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3150–3158, 2016.

[12] Lee, Y. J.; Kim, J.; Grauman, K. Key-segments for video object segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, 1995–2002, 2011.

[13] Papazoglou, A.; Ferrari, V. Fast object segmentation in unconstrained video. In: Proceedings of the IEEE International Conference on Computer Vision, 1777–1784, 2013.

[14] Yang, Y. C.; Sundaramoorthi, G.; Soatto, S. Self-occlusions and disocclusions in causal video object segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, 4408–4416 2015.

[15] Colombari, A.; Fusiello, A.; Murino, V. Segmentation and tracking of multiple video objects. *Pattern Recognition* Vol. 40, No. 4, 1307–1317, 2007.

[16] Li, F. X.; Kim, T.; Humayun, A.; Tsai, D.; Rehg, J. M. Video segmentation by tracking many figure-ground segments. In: Proceedings of the IEEE International Conference on Computer Vision, 2192–2199, 2013.

[17] Seguin, G.; Bojanowski, P.; Lajugie, R.; Laptev, I. Instance-level video segmentation from object tracks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3678–3687, 2016.

[18] Drayer B.; Brox, T. Object detection, tracking, and motion segmentation for object-level video segmentation.*arXiv preprint* arXiv:1608.03066, 2016.

[19] Pont-Tuset, J.; Perazzi, F.; Caelles, S.; Arbeláez, P.; Sorkine-Hornung, A.; van Gool, L. The 2017 DAVIS challenge on video object segmentation. *arXiv preprint* arXiv:1704.00675, 2017.

[20] Voigtlaender, P.; Leibe, B. Online adaptation of convolutional neural networks for the 2017 DAVIS challenge on video object segmentation. In: Proceedings of the DAVIS Challenge on Video Object Segmentation, 2017.

[21] Khoreva, A.; Benenson, R.; Ilg, E.; Brox, T.; Schiele, B. Lucid data dreaming for object tracking. In: Proceedings of the DAVIS Challenge on Video Object Segmentation, 2017.

[22] Tokmakov, P.; Alahari, K.; Schmid, C. Learning motion patterns in videos. In: Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition, 3386–3394, 2017.

[23] Li, X.; Qi, Y.; Wang, Z.; Chen, K.; Liu, Z.; Shi, J.; Luo, P.; Tang, X.; Loy, C. C. Video object segmentation with re-identification. In: Proceedings of the DAVIS Challenge on Video Object Segmentation, 2017.

[24] Hu, Y.-T.; Huang, J.-B.; Schwing, A. MaskRNN: Instance level video object segmentation. In: Proceedings of the 31st Conference on Neural Information Processing Systems, 324–333, 2017.

[25] Xu, N.; Yang, L.; Fan, Y.; Yue, D.; Liang, Y.; Yang, J.; Huang, T. YouTube-VOS: A large-scale video object segmentation benchmark. *arXiv preprint* arXiv:1809.03327, 2018.

[26] Luiten, J.; Voigtlaender, P.; Leibe, B. PReMVOS: Proposal-generation, refinement and merging for video object segmentation. In: *Computer Vision – ACCV 2018. Lecture Notes in Computer Science, Vol. 11364*. Jawahar, C.; Li, H.; Mori, G.; Schindler, K. Eds. Springer Cham, 565–580, 2019.

[27] Caelles, S.; Montes, A.; Maninis, K.-K.; Chen, Y.; van Gool, L.; Perazzi, F.; Pont-Tuset, J. The 2018 DAVIS challenge on video object segmentation. *arXiv preprint* arXiv:1803.00557, 2018.

[28] He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, 2961–2969, 2017.

[29] Xu, N.; Price, B.; Cohen, S.; Yang, J.; Huang, T. Deep grabcut for object selection. *arXiv preprint* arXiv:1707.00243, 2017.

[30] Chen, L. C.; Zhu, Y. K.; Papandreou, G.; Schroff, F.; Adam, H. Encoder–decoder with atrous separable convolution for semantic image segmentation. In: *Computer Vision – ECCV 2018. Lecture Notes in Computer Science, Vol. 11211*. Ferrari, V.; Hebert, M.; Sminchisescu, C.; Weiss, Y. Eds. Springer Cham, 833–851, 2018.

[31] Yang, L.; Wang, Y.; Xiong, X.; Yang, J.; Katsaggelos, A. K. Efficient video object segmentation via network modulation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 6499–6507, 2018.

[32] Chen, Y. H.; Pont-Tuset, J.; Montes, A.; van Gool, L. Blazingly fast video object segmentation with pixel-wise metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1189–1198, 2018.

[33] Cheng, J.; Tsai, Y.-H.; Hung, W.-C.; Wang, S.; Yang, M.-H. Fast and accurate online video object segmentation via tracking parts. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7415–7424, 2018.

[34] Oh, S. W.; Lee, J. Y.; Sunkavalli, K.; Kim, S. J. Fast video object segmentation by reference-guided mask propagation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 7376–7385, 2018.

[35] Leake, M.; Davis, A.; Truong, A.; Agrawala, M. Computational video editing for dialogue-driven scenes. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 130, 2017.

[36] Cui, Z. P.; Wang, O.; Tan, P.; Wang, J. Time slice video synthesis by robust video alignment. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 131, 2017.

[37] Bonneel, N.; Sunkavalli, K.; Tompkin, J.; Sun, D. Q.; Paris, S.; Pfister, H. Interactive intrinsic video editing. *ACM Transactions on Graphics* Vol. 33, No. 6, 1–10, 2014.

[38] Zhang, F. L.; Wu, X.; Zhang, H. T.; Wang, J.; Hu, S. M. Robust background identification for dynamic video editing. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 197, 2016.

[39] Levin, A.; Lischinski, D.; Weiss, Y. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 30, No. 2, 228–242, 2008.

[40] Chuang, Y. Y.; Agarwala, A.; Curless, B.; Salesin, D. H.; Szeliski, R. Video matting of complex scenes. *ACM Transactions on Graphics* Vol. 21, No. 3, 243–248, 2002.

[41] Aksoy, Y.; Oh, T. H.; Paris, S.; Pollefeys, M.; Matusik, W. Semantic soft segmentation. *ACM Transactions on Graphics* Vol. 37, No. 4, Article No. 72, 2018.

[42] Masnou, S.; Morel, J.-M. Level lines based disocclusion. In: Proceedings of the International Conference on Image Processing, Vol. 3, 259–263, 1998.

[43] Bertalmio, M.; Sapiro, G.; Caselles, V.; Ballester, C. Image inpainting. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, 417–424, 2000.

[44] Drori, I.; Cohen-Or, D.; Yeshurun, H. Fragment-based image completion. *ACM Transactions on Graphics* Vol. 22, No. 3, 303–312, 2003.

[45] Criminisi, A.; Perez, P.; Toyama, K. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing* Vol. 13, No. 9, 1200–1212, 2004.

[46] Efros, A. A.; Leung, T. K. Texture synthesis by non-parametric sampling. In: Proceedings of the 7th IEEE International Conference on Computer Vision, 1033–1038, 1999.

[47] Patwardhan, K. A.; Sapiro, G.; Bertalmio, M. Video inpainting of occluding and occluded objects. In: Proceedings of the IEEE International Conference on Image Processing, II-69, 2005.

[48] Patwardhan, K. A.; Sapiro, G.; Bertalmio, M. Video inpainting under constrained camera motion. *IEEE Transactions on Image Processing* Vol. 16, No. 2, 545–553, 2007.

[49] Granados, M.; Kim, K. I.; Tompkin, J.; Kautz, J.; Theobalt, C. Background inpainting for videos with dynamic objects and a free-moving camera. In: *Computer Vision – ECCV 2012. Lecture Notes in Computer Science, Vol. 7572*. Fitzgibbon, A.; Lazebnik, S.; Perona, P.; Sato, Y.; Schmid, C. Eds. Springer Berlin Heidelberg, 682–695, 2012.

[50] Herling, J.; Broll, W. PixMix: A real-time approach to high-quality diminished reality. In: Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, 141–150, 2012.

[51] Jia, J. Y.; Tai, Y. W.; Wu, T. P.; Tang, C. K. Video repairing under variable illumination using cyclic motions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 28, No. 5, 832–839, 2006.

[52] Grossauer, H. Inpainting of movies using optical flow. In:*Mathematical Models for Registration and Applications to Medical Imaging. Mathematics in Industry, Vol. 10*. Scherzer, O. Ed. Springer Berlin Heidelberg, 151–162, 2006.

[53] Matsushita, Y.; Ofek, E.; Ge, W. N.; Tang, X. O.; Shum, H. Y. Full-frame video stabilization with motion inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 28, No. 7, 1150–1163, 2006.

[54] Shiratori, T.; Matsushita, Y.; Tang, X.; Kang, S. B. Video completion by motion field transfer. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 411–418, 2006.

[55] Tang, N. C.; Hsu, C. T.; Su, C. W.; Shih, T. K.; Liao, H. Y. M. Video inpainting on digitized vintage films via maintaining spatiotemporal continuity. *IEEE Transactions on Multimedia* Vol. 13, No. 4, 602–614, 2011.

[56] You, S.; Tan, R. T.; Kawakami, R.; Ikeuchi, K. Robust and fast motion estimation for video completion. In: Proceedings of the MVA, 181–184, 2013.

[57] Bokov, A.; Vatolin, D. 100+ times faster video completion by optical-flow-guided variational refinement. In: Proceedings of the 25th IEEE International Conference on Image Processing 2122–2126, 2018.

[58] Wexler, Y.; Shechtman, E.; Irani, M. Space-time completion of video. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 29, No. 3, 463–476, 2007.

[59] Newson, A.; Almansa, A.; Fradet, M.; Gousseau, Y.; Pérez, P. Video inpainting of complex scenes. *SIAM Journal on Imaging Sciences* Vol. 7, No. 4, 1993–2019, 2014.

[60] Granados, M.; Tompkin, J.; Kim, K.; Grau, O.; Kautz, J.; Theobalt, C. How not to be seen—Object removal from videos of crowded scenes. *Computer Graphics Forum* Vol. 31, No. 2pt1, 219–228, 2012.

[61] Huang, J. B.; Kang, S. B.; Ahuja, N.; Kopf, J. Temporally coherent completion of dynamic video. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 196, 2016.

[62] Le, T. T.; Almansa, A.; Gousseau, Y.; Masnou, S. Demonstration abstract: Motion-consistent video inpainting. In: Proceedings of the IEEE International Conference on Image Processing, 4587, 2017.

[63] Pathak, D.; Krahenbuhl, P.; Donahue, J.; Darrell, T.; Efros, A. A. Context encoders: Feature learning by inpainting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2536–2544, 2016.

[64] Iizuka, S.; Simo-Serra, E.; Ishikawa, H. Globally and locally consistent image completion. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 107, 2017.

[65] Vo, H. V.; Duong, N. Q. K.; Pérez, P. Structural inpainting. In: Proceedings of the 26th ACM International Conference on Multimedia, 1948–1956, 2018.

[66] Jain, S. D.; Grauman, K. Click carving: Segmenting objects in video with point clicks. In: Proceedings of the 4th AAAI Conference on Human Computation and Crowdsourcing, 89–98, 2016.

[67] Xie, S. N.; Tu, Z. W. Holistically-nested edge detection. *International Journal of Computer Vision* Vol. 125, Nos. 1–3, 3–18, 2017.

[68] Meyer, F. Topographic distance and watershed lines. *Signal Processing* Vol. 38, No. 1, 113–125, 1994.

[69] Perazzi, F.; Pont-Tuset, J.; McWilliams, B.; van Gool, L.; Gross, M.; Sorkine-Hornung, A. A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 724–732, 2016.

[70] Yang, H. X.; Shao, L.; Zheng, F.; Wang, L.; Song, Z. Recent advances and trends in visual tracking: A review. *Neurocomputing* Vol. 74, No. 18, 3823–3831, 2011.

[71] Ramakanth, S. A.; Babu, R. V. SeamSeg: Video object segmentation using patch seams. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 376–383, 2014.

[72] Tsai, Y. H.; Yang, M. H.; Black, M. J. Video segmentation via object flow. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3899–3908, 2016.

[73] Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3431–3440, 2015.

[74] Chen, L. C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A. L. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 40, No. 4, 834–848, 2018.

[75] Everingham, M.; Eslami, S. M. A.; van Gool, L.; Williams, C. K. I.; Winn, J.; Zisserman, A. The Pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision* Vol. 111, No. 1, 98–136, 2015.

[76] Newson, A.; Almansa, A.; Gousseau, Y.; Pérez, P. Non-local patch-based image inpainting. *Image Processing on Line* Vol. 7, 373–385, 2017.

[77] Pérez, P.; Gangnet, M.; Blake, A. Poisson image editing. *ACM Transactions on Graphics* Vol. 22, No. 3, 313–318, 2003.

[78] Korman, S.; Avidan, S. Coherency sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 38, No. 6, 1099–1112, 2016.

[79] Ramakanth, S. A.; Babu, R. V. FeatureMatch: A general ANNF estimation technique and its applications. *IEEE Transactions on Image Processing* Vol. 23, No. 5, 2193–2205, 2014.

[80] Dehghan, A.; Assari, S. M.; Shah, M. GMMCP tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4091–4099, 2015.

[81] Roshan Zamir, A.; Dehghan, A.; Shah, M. GMCP-tracker: Global multi-object tracking using generalized minimum clique graphs. In: *Computer Vision – ECCV 2012. Lecture Notes in Computer Science, Vol. 7573*. Fitzgibbon, A.; Lazebnik, S.; Perona, P.; Sato, Y.; Schmid, C. Eds. Springer Berlin Heidelberg, 343–356, 2012.

[82] Bay, H.; Ess, A.; Tuytelaars, T.; van Gool, L. Speeded-up robust features (SURF). *Computer Vision and Image Understanding* Vol. 110, No. 3, 346–359, 2008.

[83] Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; Brox, T. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1647–1655, 2017.

[84] Xu, B. B.; Pathak, S.; Fujii, H.; Yamashita, A.; Asama, H. Spatio-temporal video completion in spherical image sequences. *IEEE Robotics and Automation Letters* Vol. 2, No. 4, 2032–2039, 2017.

[85] Odobez, J. M.; Bouthemy, P. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation* Vol. 6, No. 4, 348–365, 1995.

[86] Sánchez, J. Comparison of motion smoothing strategies for video stabilization using parametric models. *Image Processing on Line* Vol. 7, 309–346, 2017.

[87] Choi, S.; Kim, T.; Yu, W. Robust video stabilization to outlier motion using adaptive RANSAC. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 1897–1902, 2009.

[88] Chiu, W. C.; Fritz, M. Multi-class video co-segmentation with a generative multi-video model. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 321–328, 2013.

[89] Yang, M. Y.; Reso, M.; Tang, J.; Liao, W. T.; Rosenhahn, B. Temporally object-based video co-segmentation. In: *Advances in Visual Computing. Lecture Notes in Computer Science, Vol. 9474*. Bebis, G. et al. Eds. Springer Cham, 198–209, 2015.

Thuc Trinh Le is a Ph.D. candidate in computer science and applied mathematics at the LTCI Lab of Telecom ParisTech, Paris-Saclay University, France. His research is devoted to the development of machine learning techniques to address advanced problems in video editing, video segmentation, and video reconstruction.

Andrés Almansa has been a CNRS Research Director at Université Paris Descartes (France) since 2016. He received his M.Sc. and Ph.D. degrees from ENS Cachan (1999, 2002), and M.Sc. and Engineering degrees from Universidad de la República (1995, 1998). He has previously worked at Telecom ParisTech, ENS Cachan (France), Universitat Pompeu Fabra (Spain), and Universidad de la República (Uruguay). His current research interests include image restoration and analysis, subpixel stereovision and applications to earth observation, high quality digital photography, and film editing and restoration.

Yann Gousseau received his engineering degree from the cole Centrale de Paris, France, in 1995, and Ph.D. degree in applied mathematics from the University of Paris-Dauphine in 2000. He is currently a professor at Telecom ParisTech. His research interests include mathematical modeling of natural images and textures, stochastic geometry, computational photography, computer vision, and image and video processing.

Simon Masnou is a professor in mathematics at Claude-Bernard Lyon 1 University (France) and head of the Camille Jordan Institute. His research interests include image processing, shape optimization, calculus of variations, and geometric measure theory.