Noname manuscript No.
(will be inserted by the editor)

Patch-based Detection of Dynamic Objects in CrowdCam Images

Gagan Kanojia · Shanmuganathan Raman

Received: date / Accepted: date

Abstract A scene can be divided into two parts: static and dynamic. The parts of the scene which do not admit any motion are static regions while moving objects correspond to dynamic regions. In this work, we tackle the challenging task of identifying dynamic objects present in the CrowdCam images. Our approach exploits the coherency present in the natural images and utilizes the epipolar geometry present between a pair of images to achieve this objective. It does not require a dynamic object to be present in all the given images. We show that the proposed approach obtains state-of-the-art accuracy on standard datasets.

Keywords Object detection · Dynamic objects · Epipolar geometry

1 Introduction

Dynamic objects detection has been an active area of research for a long time. The moving objects present in the scene hold essential information about the scene. This information is used in various applications like action recognition, pedestrian detection, object tracking, and 3D modeling of moving objects. Traditionally, these tasks have been performed on video sequences. The process is easier on video sequences as spatiotemporal information is provided, but computationally more expensive. Subsequently, researchers have moved on to perform these tasks using sparse samples of these videos, i.e., images captured at certain intervals. We call them an image sequence. An image sequence does not require a lot of memory to store, transmit, and process. However, they pose certain challenges in matching, occlusion, and deformations. Temporally speaking, they are more difficult to process than videos in which the spatiotemporal information is provided [27].

As explained in [10], CrowdCam images are the images of a scene captured by the crowd. Hence, a pair of images can have a wide-baseline, and dynamic objects can move a significant distance or even leave the scene. These facts make them more challenging than dealing with the videos. In this work, we are interested in two kinds of regions present in a scene: static and dynamic. Static regions correspond to those parts of the scene which do not admit any motion whereas dynamic regions correspond to the moving objects present in the scene. A dynamic scene consists of objects in motion. In some cases, these dynamic objects are the key sources of information [6, 25]. While in others, they are considered outliers and need to be removed [2]. In both the scenarios, the detection of dynamic objects is an essential task.

Consider images of a dynamic scene captured using a handheld camera from different (or same) viewpoints. Since we are dealing with CrowdCam images, image pairs may have a wide-baseline. Due to dynamic objects and change in viewpoints, these images will not be aligned. That will make it difficult to decide whether a certain region is static or dynamic. Moreover, in these images, dynamic objects do not satisfy the epipolar constraint. This constraint has been exploited in previous works for the detection of dynamic objects [10]. In such scenarios, the estimation of the fundamental matrix can be noisy due to inaccurate matchings. Finding dense

Gagan Kanojia
Indian Institute of Technology Gandhinagar, Palaj, Gandhinagar, 382355, India
E-mail: gagan.kanojia@iitgn.ac.in

Shanmuganathan Raman
Indian Institute of Technology Gandhinagar, Palaj, Gandhinagar, 382355, India
E-mail: shanmuga@iitgn.ac.in

correspondences between pixel locations of a pair of images are liable to errors as well. For these reasons, relying only on one of them can lead to a significant lowering in the quality of obtained results. In this work, we present a patch-based technique which provides a binary map, differentiating the static and dynamic regions present in CrowdCam images. We combine the information provided by the fundamental matrices and the dense correspondences to improve the quality of results. We do not put any restriction on the movement of dynamic objects across the images. Furthermore, we do not restrict dynamic objects to be present in all the images.

The primary contributions of our work are as follows:

1. We develop an algorithm which achieves state-of-the-art results on the detection of dynamic objects in CrowdCam images.
2. We exploit the image coherency present in natural scenes along with the geometric relation between the pair of images.
3. We show that even without using any geometric information and learning techniques, we can detect the dynamic objects.

The rest of the paper is organized as follows. Section 2 discusses the relevant works. Section 3 describes our approach to detect dynamic objects present in the CrowdCam images. Section 4 discusses the results obtained using our approach and their comparison with a state-of-the-art method. Finally, Section 5 presents the conclusion and future challenges ahead.

2 Related Works

Various approaches have been proposed to detect dynamic objects in different scenarios like object tracking [3, 8, 34, 29], photosequencing [6, 25], and motion segmentation [33, 40, 29]. In [33], optical flow was estimated between the two images and then normalized cuts was applied to segment dynamic objects. Later, a joint estimation of optical flow and object segmentation using variational methods was proposed [8, 9]. In [22], a variant of optical flow is used to detect the lip event. Recently, some convolutional neural network based techniques have shown state-of-the-art results for such problems [13, 47, 26, 32]. A multitask deep network is proposed to jointly detect humans and estimate their head pose [46]. A survey of such techniques can be found in [44]. However, these techniques work only on video sequences. In the case of video sequences, temporal information can be exploited to find good matches between the frames which can be propagated further to detect the changes. In [45], the authors claim that the proposed optical flow technique can handle large displacements. However, our objective is not to track dynamic objects in the given sequence. It is to identify the dynamic regions present in the scene.

The other approach to detect dynamic objects is through estimating the 3D structure of the scene given multiple images captured using handheld cameras. In [41], the dynamic scene is reconstructed using a pair of images. Since a pair of images of a dynamic scene can not be related with a single projective transformation technique, the authors of [41] have segmented dynamic objects along with their 3D reconstruction using multiple motion models. This method relies on the estimation of correspondences between dynamic objects while our algorithm depends on the estimation of correspondences between static parts of the scene which are easier to obtain. We have made no such assumption that dynamic objects have to be present in multiple images. The works proposed in [10, 16] are the most relevant to our work. In [16], the authors have segregated the correspondences among dynamic objects and static regions. However, they can not handle multiple dynamic objects. The approach proposed in [10] overcomes this drawback and can detect multiple dynamic objects present in the scene. They have proposed the concept of the epipolar patch in their work which utilizes the epipolar geometry present between a pair of images. There are certain drawbacks of using epipolar geometry when dealing with natural scenes due to the recurrence of patches [48]. A patch on a dynamic object could get matched to a similar patch somewhere along the epipolar line, or the object could have moved along the epipolar line. In [11], the authors make use of the epipolar geometry to predict the location of dynamic objects in an image sequence.

3 Proposed Approach

Given a set of N images of size $m \times n$ of a dynamic scene, our objective is to detect dynamic objects present in each image. An image I_r is chosen as the reference image in which we want to detect dynamic objects. Among other $N-1$ images, those having sufficient overlap with the reference image for the estimation of fundamental matrices, are considered as source images. Let us assume that there are k source images for the reference image I_r. The approach for finding dynamic objects is the same for each image. Hence, we demonstrate it for one image. In this paper, we consider the top-left corner of an image as the origin and coordinates increase as we move towards right and bottom respectively.

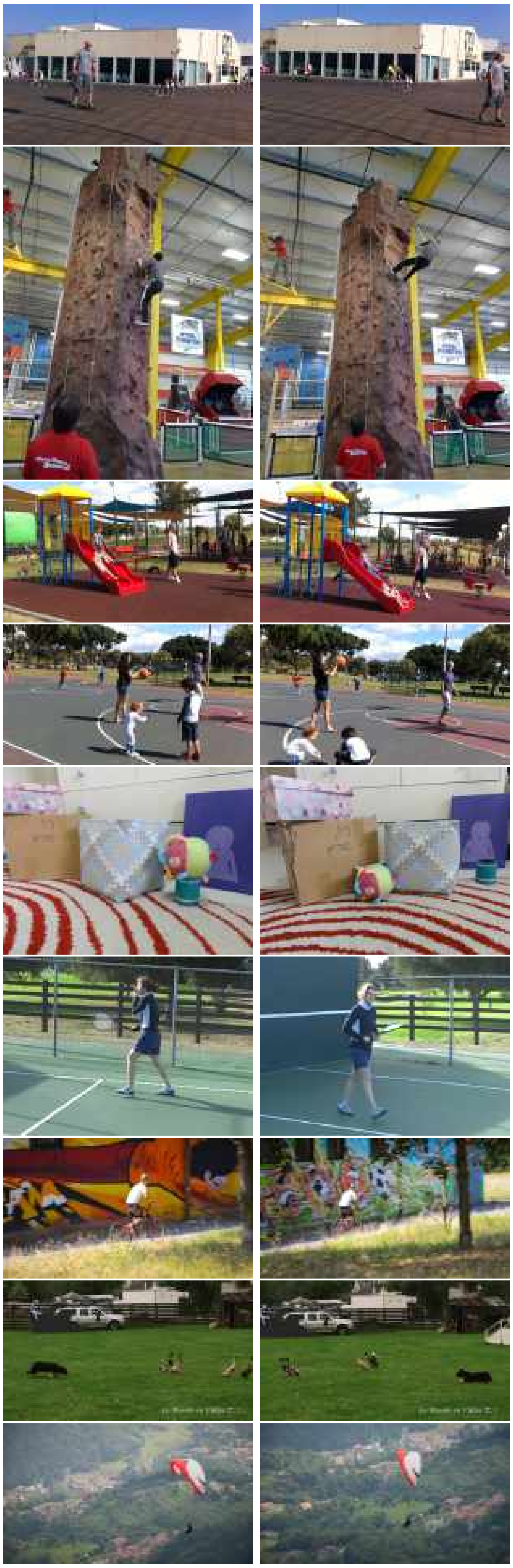

Fig. 1 The figure shows 2 images each from some datasets used in this work.

3.1 Fundamental Matrices

A set of fundamental matrices $\mathcal{F}$ is computed such that

$$\mathcal{F} = \bigcup_{s=1}^{k} F_s \tag{1}$$

where, F_s is the fundamental matrix estimated between the reference image I_r and the source image I_s. There are many efficient ways to estimate a fundamental matrix [15, 18]. However, in the case of dynamic scenes, estimation of fundamental matrices are often noisy. Hence, we can not rely entirely on them.

3.2 Confidence Map

The dense correspondence map $\mathcal{N}_{r\rightarrow s} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, is estimated for the reference image I_r with each source image I_s, where $s = 1, 2, \ldots, k$. Finding a dense set of correspondences between two images has been an active area of research for a long time. Many approaches have been proposed to address this problem. Among these approaches, Generalized PatchMatch (GPM) [5], Coherent Sensitive Hashing (CSH) [20], NRDC [17], Sift Flow [21], and DeepFlow [43] are the most well known. In this work, we are exploiting the coherency present in the images which motivated the use of Generalized PatchMatch [5] as it exploits the coherency to find the correspondences. CSH also exploits the coherency present in the scene and NRDC algorithm uses PatchMatch [4] for initialization. Hence, they can also be used to find dense correspondences. However, apart from exploiting coherency, Generalized PatchMatch also allows us to match arbitrary descriptors for finding the correspondences. It allows us to find the correspondences using arbitrary features instead of just using the color values. In this work, we have computed the dense correspondences by applying Generalized PatchMatch on the feature vectors which are computed for every patch at each location of the reference image I_r. To quantify the measure of confidence associated with each correspondence, we use the following confidence map $Conf_{(s)} : \mathbb{R}^2 \rightarrow \mathbb{R}$ for each pair (I_r, I_s), where $s = 1, \ldots, k$.

$$Conf_{(s)}(i,j) = S(\|f_r(i,j) - f_s(\mathcal{N}_{r\rightarrow s}(i,j))\|_2) \tag{2}$$

where, $i = 1, \ldots, m$, $j = 1, \ldots, n$, $\|.\|_2$ denotes the ℓ_2-norm of the vector, and $S(x) = e^{-\frac{x^2}{2\sigma_1^2}}$. Here, f_r and f_s map the locations of the reference image I_r and the source image I_s to the corresponding feature vectors, respectively. Hence, $f_r(i,j)$ is the feature vector at the location (i,j) in the reference image, $f_s(\mathcal{N}_{r\rightarrow s}(i,j))$ is the feature vector at the corresponding location of (i, j) in the source image I_s. We will see in section 3.3.2, that the decision of whether a patch is static or dynamic depends on the clustering of feature vectors corresponding to the contender patch locations (section 3.3.1). Hence, by finding dense correspondence using feature vectors, Generalized PatchMatch allows us to be consistent with the notion of similarity throughout the algorithm. In this work, we have used VLFeat implementation of densely sampled SIFT features [23, 38].

3.3 Patch-based Detection of Dynamic Objects

We compute the labels, i.e., static or dynamic, at each pixel location of the reference image in two orders. First, from the top-left to the bottom-right corner in row-major order and the second, in the reverse order, i.e., from the bottom-right to the top-left corner. At each pixel location, we identify the contender patch locations (explained in 3.3.1) for the current location $\boldsymbol{x}_r$ in the reference image. Then, based on the contender patch locations, the confidence maps and the fundamental matrices estimated between the reference image and the source images, we decide the label, i.e., static or dynamic, of $\boldsymbol{x}_r$ (explained in 3.3.2). Finally, if $\boldsymbol{x}_r$ is labeled as dynamic, we update the dense correspondence map $\mathcal{N}_{r\to s}(\boldsymbol{x}_r)$ so that $\boldsymbol{x}_r$ is mapped to the probable corresponding location of the occluded static region at $\boldsymbol{x}_r$, in the source images (explained in 3.3.3). Algorithm 1 shows the pseudo-code for the proposed approach.

3.3.1 Contender Patch Locations

To decide whether a patch location $\boldsymbol{x}_r = [i, j]^\top$, where i and j are the x and y coordinates in the reference image, belongs to static or dynamic region in the scene, we carefully select some locations in the source images. We call these selected locations as the contender patch locations C. If $\boldsymbol{x}_r$ belongs to the dynamic object, then these locations are the probable candidates for the nearest neighbors of the occluded static region at $\boldsymbol{x}_r$ in the source images. First, we discuss the scan for top-left to bottom-right corner. Let

$$\hat{\boldsymbol{x}}_1^{\downarrow} = (i-1, j) \tag{3}$$

$$\hat{\boldsymbol{x}}_2^{\downarrow} = (i, j-1) \tag{4}$$

where, $\hat{\boldsymbol{x}}_1^{\downarrow}$ and $\hat{\boldsymbol{x}}_2^{\downarrow}$ are the locations on left and above the current location $\boldsymbol{x}_r$ in the reference image I_r, respectively, and

$$\boldsymbol{x}_{1,s}^{\downarrow} = (x_{1,s}^{\downarrow}, y_{1,s}^{\downarrow}) = \mathcal{N}_{r\to s}(\hat{\boldsymbol{x}}_1^{\downarrow}) \tag{5}$$

$$\boldsymbol{x}_{2,s}^{\downarrow} = (x_{2,s}^{\downarrow}, y_{2,s}^{\downarrow}) = \mathcal{N}_{r\to s}(\hat{\boldsymbol{x}}_2^{\downarrow}) \tag{6}$$

Algorithm 1 Patch-based Detection of Dynamic Objects

```
Input: Reference Image $I_r$, Source Images $\{I_1, \dots I_k\}$
Output: Binary Mask $label : \mathbb{R}^2 \to \{0,1\}$
for $s = 1 \to k$ do
    Find matches between $I_r$ and $I_s$
    Estimate fundamental matrix $F_s : I_r \to I_s$
    Compute dense correspondence map $\mathcal{N}_{r\to s}$
    Initialize $\mathcal{N}_{r\to s}^{\uparrow} = \mathcal{N}_{r\to s}^{\downarrow} = \mathcal{N}_{r\to s}$
    Compute the confidence Map $Conf_{(s)}$ (Section 3.2)
    Initialize $Conf_{(s)}^{\uparrow} = Conf_{(s)}^{\downarrow} = Conf_{(s)}$
end for
for $h \in \{\downarrow, \uparrow\}$ do
    for $i = 1 \to m$ do
        for $j = 1 \to n$ do
            Find $C^h$ for $\boldsymbol{x}_r = [i,j]^\top$ (Section 3.3.1)
            Construct $\mathcal{A}$ using $C^h$ (Eq: 14,15)
            Apply DBSCAN on $\mathcal{A}$
            Find $\bar{f}$ for the cluster which has the maximum
            value of $c_k$ (Eq: 17,16)
            Find the values for $l$ and $g$ (Eq: 18,19)
            Decide $label^h(i,j)$ on the basis of $P(\boldsymbol{x}_r)$ (Eq: 20)
            if $label^h(i,j) == 1$ then
                Update $\mathcal{N}_{r\to s}^{h}(\boldsymbol{x}_r)$ and $Conf_{(s)}^{h}(\boldsymbol{x}_r)$,
                $\forall s = 1, \dots, k$ (Eq: 22,23)
            end if
        end for
    end for
end for
$label = label^{\downarrow} \odot label^{\uparrow}$, where $\odot$ denotes the element-wise
multiplication.
```

where, $\boldsymbol{x}_{1,s}^{\downarrow}$ and $\boldsymbol{x}_{2,s}^{\downarrow}$ are the locations of the nearest neighbors of the patches on the patch locations $\hat{\boldsymbol{x}}_1^{\downarrow}$ and $\hat{\boldsymbol{x}}_2^{\downarrow}$ in the source image I_s, respectively.

$$C_s^{\downarrow} = \{C_{1,s}^{\downarrow}, C_{2,s}^{\downarrow}\} = \{(x_{1,s}^{\downarrow}+1, y_{1,s}^{\downarrow}), (x_{2,s}^{\downarrow}, y_{2,s}^{\downarrow}+1)\} \tag{7}$$

Here, $C_s^{\downarrow}$ is the set of locations on the right of $\boldsymbol{x}_{1,s}^{\downarrow}$ and below of $\boldsymbol{x}_{2,s}^{\downarrow}$ in the source image I_s, respectively. Similarly, in the reverse scan, let

$$\hat{\boldsymbol{x}}_1^{\uparrow} = (i+1, j) \tag{8}$$

$$\hat{\boldsymbol{x}}_2^{\uparrow} = (i, j+1) \tag{9}$$

where, $\hat{\boldsymbol{x}}_1^{\uparrow}$ and $\hat{\boldsymbol{x}}_2^{\uparrow}$ are the locations on right and below the current location $\boldsymbol{x}_r$ in the reference image I_r respectively, and

$$\boldsymbol{x}_{1,s}^{\uparrow} = (x_{1,s}^{\uparrow}, y_{1,s}^{\uparrow}) = \mathcal{N}_{r\to s}(\hat{\boldsymbol{x}}_1^{\uparrow}) \tag{10}$$

$$\boldsymbol{x}_{2,s}^{\uparrow} = (x_{2,s}^{\uparrow}, y_{2,s}^{\uparrow}) = \mathcal{N}_{r\to s}(\hat{\boldsymbol{x}}_2^{\uparrow}) \tag{11}$$

where, $\boldsymbol{x}_{1,s}^{\uparrow}$ and $\boldsymbol{x}_{2,s}^{\uparrow}$ are the locations of the nearest neighbors of the patches on the patch locations $\hat{\boldsymbol{x}}_1^{\uparrow}$ and $\hat{\boldsymbol{x}}_2^{\uparrow}$ in the source image I_s, respectively.

$$C_s^{\uparrow} = \{C_{1,s}^{\uparrow}, C_{2,s}^{\uparrow}\} = \{(x_{1,s}^{\uparrow}-1, y_{1,s}^{\uparrow}), (x_{2,s}^{\uparrow}, y_{2,s}^{\uparrow}-1)\} \tag{12}$$

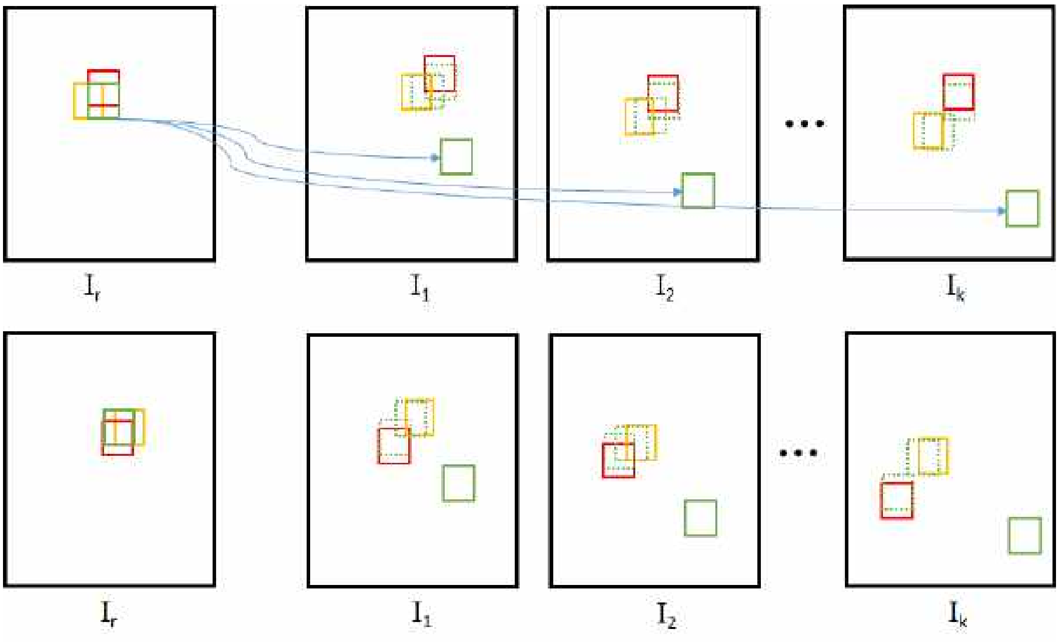

Fig. 2 The figure illustrates the computation of the contender patch locations given the reference image I_r and source images $I_1, I_2, \ldots, I_k$. The first and the second row show the computation of contender patch locations in top-left to bottom-right ($\downarrow$) and bottom-right to top-left ($\uparrow$) scan, respectively. The reference patch at $\boldsymbol{x}_r = [i, j]^\top$ is shown with the green color in the reference image I_r and its corresponding patch locations in the source images $\{I_s\}_{s=1}^k$ are also shown with the same color. In the first row, the patches at $\hat{\boldsymbol{x}}_1^\downarrow$ and $\hat{\boldsymbol{x}}_2^\downarrow$ in the I_r are shown with yellow and red color, respectively, and their corresponding patch locations $\boldsymbol{x}_{1,s}^\downarrow$ and $\boldsymbol{x}_{2,s}^\downarrow$ in the source images $\{I_s\}_{s=1}^k$ are also shown with the same colors. Similarly, in the second row, the patches at $\hat{\boldsymbol{x}}_1^\uparrow$ and $\hat{\boldsymbol{x}}_2^\uparrow$ in the I_r are shown with yellow and red color, respectively, and their corresponding patch locations $\boldsymbol{x}_{1,s}^\uparrow$ and $\boldsymbol{x}_{2,s}^\uparrow$ in the source images $\{I_s\}_{s=1}^k$ are also shown with the same colors. The contender patch locations C_s^h in the image I_s are shown with the green box with the dotted edges, where $h \in \{\downarrow, \uparrow\}$ and $s = 1, 2, \ldots k$. In this figure, the reference patch belongs to the dynamic region.

Here, $C_s^\uparrow$ is the set of locations on the left of $\boldsymbol{x}_{1,s}^\uparrow$ and above of $\boldsymbol{x}_{2,s}^\uparrow$ in the source image I_s, respectively.

$$C^h = \bigcup_{s=1}^{k} C_s^h \tag{13}$$

where, $h \in \{\downarrow, \uparrow\}$. Here, $\downarrow$ and $\uparrow$ signify the scan from top-left to bottom-right and bottom-right to top-left, respectively. The variables with $\downarrow$ as the superscript are computed during the top-left to bottom-right scan, while the variables with $\uparrow$ as the superscript are computed during the bottom-right to top-left scan. If there are k source images, then there would be $2k$ contender patch locations. Fig. 2 illustrates the computation of the contender patch locations given the reference image I_r and the source images I_s, where $s = 1, 2, \ldots, k$. By exploiting the coherency present in natural scenes, we can consider that C_s^h is the set of the probable nearest neighbor patch locations of $\boldsymbol{x}_r$ in source image I_s. For example: if $x_{L_{(s)}}$ is the nearest neighbor of the patch on the left of $\boldsymbol{x}_r$, then the patch on the right of $x_{L_{(s)}}$ should be the nearest neighbor of $\boldsymbol{x}_r$. However, in the case of CrowdCam images, this will be true for the planar static regions but not on the boundaries of objects. This would be dealt using the fundamental matrices as explained in 3.3.2. After deciding the contender patch locations, the rest of the steps are independent of whether we are going from top-left to bottom-right or vice versa. Hence, we will drop the $\{\uparrow, \downarrow\}$ symbols in the upcoming sections.

3.3.2 Labelling

The contender patch locations of $\boldsymbol{x}_r$ correspond either to the location of the static region corresponding to the $\boldsymbol{x}_r$ or some other (or the same) dynamic object occluding the static region in the corresponding source image. Being a probabilistic approach, it is assumed that the static region corresponding to $\boldsymbol{x}_r$ is not occluded in a sufficient number of images.

$$\mathcal{A}_s = \{\boldsymbol{f}_{1,s}, \boldsymbol{f}_{2,s}\} = \{f_s(C_{1,s}), f_s(C_{2,s})\} \tag{14}$$

Here, f_s maps the locations of the source image I_s to the corresponding feature vectors (section 3.2) and $\mathcal{A}_s$ is the set of feature vectors $\boldsymbol{f}_{1,s}$ and $\boldsymbol{f}_{2,s}$ corresponding to the contender patch locations $C_{1,s}$ and $C_{2,s}$ in the image I_s.

$$\mathcal{A} = \bigcup_{i=1}^{k} \mathcal{A}_s \tag{15}$$

Here, $\mathcal{A}$ is the set of feature descriptors corresponding to the contender patch locations. For feature descriptors, there can be many choices. The simplest choice would be to take a patch of $p \times p$ at each contender patch location. Then, either vectorize its intensity values or create a histogram of intensity values to use it as a feature vector. However, they do not perform well when it comes to matching in the images captured with significant difference in the viewpoints. In this work, we experimented with different feature descriptors like DAISY [37], OTC [24] ,and dense SIFT [23, 38]. The overall performance of dense SIFT was better in our case. Hence, in this work, we have used dense SIFT as the feature descriptor computed for a patch of size $p \times p$. We have used $p = 32$ in all our experiments.

Now, we apply DBSCAN on the feature descriptors of $\mathcal{A}$ [14]. DBSCAN is robust to outliers and in our case, we do not know the number of clusters. This is due to the fact that, apart from the static part, there can be multiple dynamic objects occluding that static part in the source images. The feature vectors corresponding to dynamic objects and the static regions will fall into different clusters. Consider that DBSCAN outputs $\varPhi$ clusters i.e., $E_1, E_2, \ldots, E_{\varPhi}$. Then, we calculate the aggregated confidence c_ϕ associated with the ϕ^{th} cluster using the confidence map.

$$c_\phi = \sum_{\boldsymbol{f}_{g,s} \in E_\phi} Conf_{(s)}(\hat{\boldsymbol{x}}_g) \tag{16}$$

where, $s = 1, \ldots, k$, $\phi \in \{1, \ldots, \varPhi\}$ and $g \in \{1, 2\}$. The reason behind using the confidence values $Conf_{(s)}(\hat{\boldsymbol{x}}_g)$ is that the confidence of a contender patch location to be a candidate for $\boldsymbol{x}_r = [i, j]$ depends on how confidently the neighboring features of the contender patch location and $\boldsymbol{x}_r$ are matched (neighbors depend on the order of the current scan). In accordance to the assumption that the static region corresponding to $\boldsymbol{x}_r$ are not occluded in a sufficient number of images, the locations belonging to the static region would be matched with high confidence and would be sufficient in number. Hence, the most confident cluster would refer to the patches belonging to the static region. Now, the weighted mean value of the features, i.e. $\bar{\boldsymbol{f}}$, of the most confident cluster is calculated.

$$\bar{\boldsymbol{f}} = \frac{1}{c_i} \sum_{\boldsymbol{f}_{g,s} \in E_i} Conf_{(s)}(\hat{\boldsymbol{x}}_g) \boldsymbol{f}_{g,s} \tag{17}$$

where, $c_i = \max(\{c_1, c_2, \ldots, c_{\varPhi}\})$ and i is the index of the maximum value. The probability of $\boldsymbol{x}_r$ to be a static location based on the contender patch locations is

$$l(\boldsymbol{x}_r, C) = e^{-\dfrac{\|f_r(x_r) - \bar{\boldsymbol{f}}\|_2^2}{2\sigma_2^2}}. \tag{18}$$

The above equation says that the probability of the reference location $\boldsymbol{x}_r$ to belong to the static region decreases as $f_r(\boldsymbol{x}_r)$ moves away from the centroid of the most confident cluster. The probability of $\boldsymbol{x}_r$ to be a static location based on the fundamental matrices is

$$g(\boldsymbol{x}_r, \boldsymbol{x}_s, \mathcal{F}) = S\left(\frac{1}{Z}\sum_{s=1}^{k} Conf_{(s)}(\boldsymbol{x}_s) d(\boldsymbol{x}_r, \boldsymbol{x}_s)\right) \tag{19}$$

where,
$\boldsymbol{x}_s = \mathcal{N}_{r \to s}(\boldsymbol{x}_r), \forall s = 1, \ldots, k,$
$Z = \sum_{s=1}^{k} Conf_{(s)}(\boldsymbol{x}_s),$
$d(\boldsymbol{x}_r, \boldsymbol{x}_s) = \dfrac{\boldsymbol{x}_s F_s \boldsymbol{x}_r^T}{(F_s \boldsymbol{x}_r^\top)_1^2 + (F_s \boldsymbol{x}_r^\top)_2^2 + (\boldsymbol{x}_s F_s)_1^2 + (\boldsymbol{x}_s F_s)_2^2},$
and $S(x) = e^{-\frac{x^2}{2\sigma_3^2}}$.

Here, $d(.,.)$ is the Sampson distance [18]. g is the weighted average of the sampson distance between the reference patch location $\boldsymbol{x}_r$ and its corresponding locations $\boldsymbol{x}_s$ in the source images $\{I_s\}_{i=1}^{k}$. If $\boldsymbol{x}_r$ belongs to the static region, then the corresponding patches will lie very close to the epipolar line corresponding to $\boldsymbol{x}_r$ in the source images. This is not necessarily true if $\boldsymbol{x}_r$ belongs to the dynamic region. As we have already pointed out in the beginning of this section, it is a probabilistic approach. Hence, the probability of $\boldsymbol{x}_r$ to belong to the static region will be more if its corresponding points in the source images lie close to epipolar lines and vice versa. Ideally, the static points should have $d(\boldsymbol{x}_r, \boldsymbol{x}_s) = 0$. However, since the estimation of fundamental matrix can be noisy, such hard constraints will not always get satisfied.

In order to decide the label of $\boldsymbol{x}_r$, we define an energy function P.

$$P(\boldsymbol{x}_r) = (1 - w(\boldsymbol{x}_r)) l(\boldsymbol{x}_r, C) + w(\boldsymbol{x}_r) g(\boldsymbol{x}_r, \boldsymbol{x}_s, \mathcal{F}) \tag{20}$$

where, $w(\boldsymbol{x}_r) \in [0, 1]$. Here, w is a function of $\boldsymbol{x}_r$ which weights the contribution of the l and g towards the decision. As pointed out in the previous section, the coherency can be exploited in the planar regions. While at the boundaries of the objects, the reasoning behind the contender patch locations may not hold. In that case, $w(\boldsymbol{x}_r)$ puts more weight on g than l. In this work, we have used the edge detection technique presented in [12]. If a patch p around $\boldsymbol{x}_r$ contains an edge, then the value of $w(\boldsymbol{x}_r)$ is w_e, otherwise its value is w_p. The label of the $\boldsymbol{x}_r$ is decided as follows:

$$label(\boldsymbol{x}_r) = \begin{cases} 0, & \text{if } P(\boldsymbol{x}_r) > th \\ 1, & \text{otherwise} \end{cases} \tag{21}$$

where, 0 stands for the static region and 1 stands for the dynamic region. Here, th is a hyper-parameter which bounds the minimum energy required for a location to belong to a static region.

3.3.3 Update

If the location $\boldsymbol{x}_r$ is labeled as static, then we move on to the next location. However, if the location $\boldsymbol{x}_r$ is labeled as dynamic, then we update its nearest neighbor mapping and confidence map as follows:

$$\mathcal{N}_{r\to s}(\boldsymbol{x}_r) = C_{g,s} \tag{22}$$

$$Conf_{(s)}(\boldsymbol{x}_r) = Conf_{(s)}(\hat{\boldsymbol{x}}_g) \tag{23}$$

where $s = 1, \ldots, k$. The value of g is such that

$$Conf_{(s)}(\hat{\boldsymbol{x}}_g) = \max(Conf_{(s)}(\hat{\boldsymbol{x}}_1), Conf_{(s)}(\hat{\boldsymbol{x}}_2))$$

and the value of i is such that $c_i = \max(\{c_\phi\}_{\phi=1}^{\Phi})$ i.e., the most confident cluster index.

After computing the binary map, $label^{\downarrow}$ and $label^{\uparrow}$ for both the scan orders respectively, the final binary map $label$ is computed as follows:

$$label = label^{\downarrow} \odot label^{\uparrow} \tag{24}$$

where $\odot$ denotes the element-wise multiplication.

4 Results and Discussion

4.1 Datasets

For the evaluation of the proposed algorithm, we have used the playground, basketball and skateboard datasets used in [6], rock-climbing dataset used in [28], toy-ball dataset captured by [10], tennis dataset used in [7], and few scenes from DAVIS dataset [31, 30]. DAVIS dataset contains the video sequences. Hence, we have picked the frames at sufficient intervals in such a way that the alternate images have sufficient overlap for the estimation of fundamental matrices while keeping the detection challenging. Out of 90, 87, 86, 35, 70, and 80 images from bmx-bumps, boxing-fixeye, dog-gooses, rollerblade, paragliding, and car-turn scenes of the DAVIS dataset, we have picked 9, 11, 8, 6, 8, and 9 images respectively. Fig. 1 shows two images each from some of the datasets used. All the datasets involve camera motion and object motion. The datasets cover the scenes which contain single or multiple dynamic objects.

Dataset	Dafni *et al.* [10]	Ours
Skateboard	0.42 ± 0.1	0.5 ± 0.007
Basketball	0.47 ± 0.04	0.51 ± 0.0004
Climbing	0.13 ± 0.05	0.34 ± 0.03
Playground	0.32 ± 0.11	0.365 ± 0.012
Toy ball	0.6 ± 0.03	0.44 ± 0.04

Table 1 The table shows the comparison between Dafni *et al.* [10] and our proposed approach in terms of Jaccard index.

4.2 Evaluation

We have compared our results with the state-of-the-art method proposed in Dafni *et al.* [10]. We have computed the Jaccard Index, which was used in [10], for our results over the datasets and provided the comparison in Table 1.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{25}$$

where, J is the Jaccard index and A and B are the two sets among which the similarity needs to be estimated. In our case, A is the ground truth mask and B is the mask obtained using our approach. We have used the VLFeat implementation of dense SIFT [23, 38] for the feature descriptors. We normalized these features by dividing them by their maximum value for the given set. The values of σ_1, σ_2, σ_3, w_e, and w_p used for all the datasets are 0.35, 0.45, 20, 0.5, and 0, respectively. The threshold used in DBSCAN for all the datasets is 0.1. The value of th is in the range of 0.5 to 0.75. We have performed the experiments with different values of the parameters and these are the values which gave the overall best performance over all the datasets. The value of th is the same for all the images in a dataset. Hence, we have compared the Jaccard index computed on our results with the Jaccard indices in [10] for the optimal threshold per set. Fig. 3 shows the comparison of results on skateboard dataset obtained in [10] with our results. It can be seen that we have obtained better true positives as compared to [10]. In Fig. 3, (a) and (b) show the binary map and corresponding highlighted region in the image obtained in [10]. In Fig. 3, (c) and (d) show the binary map and corresponding highlighted region in the image obtained using our approach. Similarly, we compare our results on some of images of basketball, climbing, toy ball, and playground datasets in Fig. 4, Fig. 6, and Fig. 5, respectively. In Fig. 6, it can be seen that not all the dynamic objects have been detected. Those dynamic objects have hardly moved in the whole image sequence which made the algorithm to consider them as the static objects. In Fig. 7 and Fig. 8, the results obtained on bmx-bumps, boxing-fixeye, rollerblade, dog-gooses, and tennis datasets using our approach are shown. The results in Fig. 7 are obtained

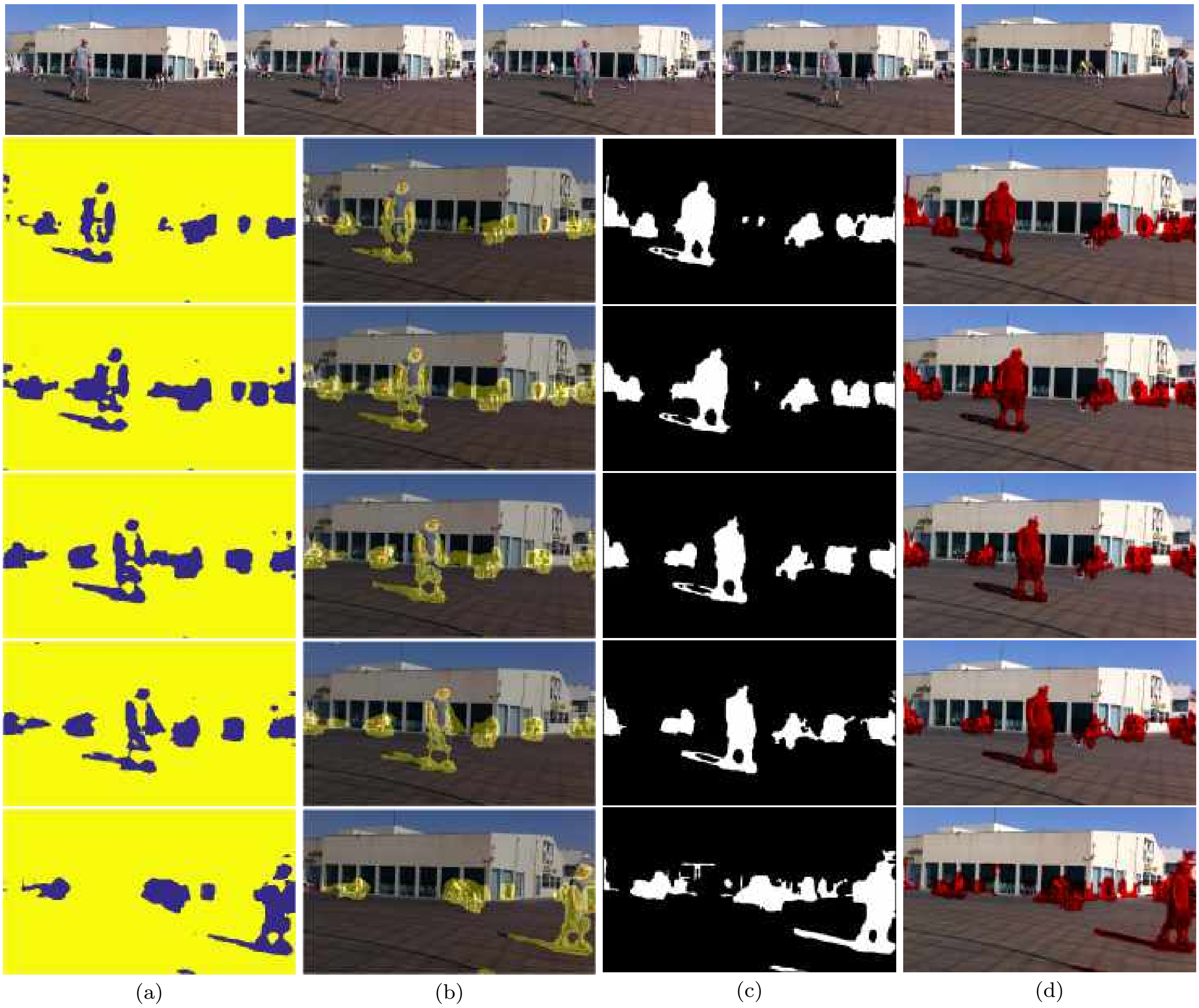

Fig. 3 The figure shows the comparison between the results obtained on skateboard dataset using [10] and our approach. The first row shows the input image stack. (a) and (b) show the mask and the corresponding highlighted region obtained in [10] while (c) and (d) show the mask and the corresponding highlighted region obtained using our approach.

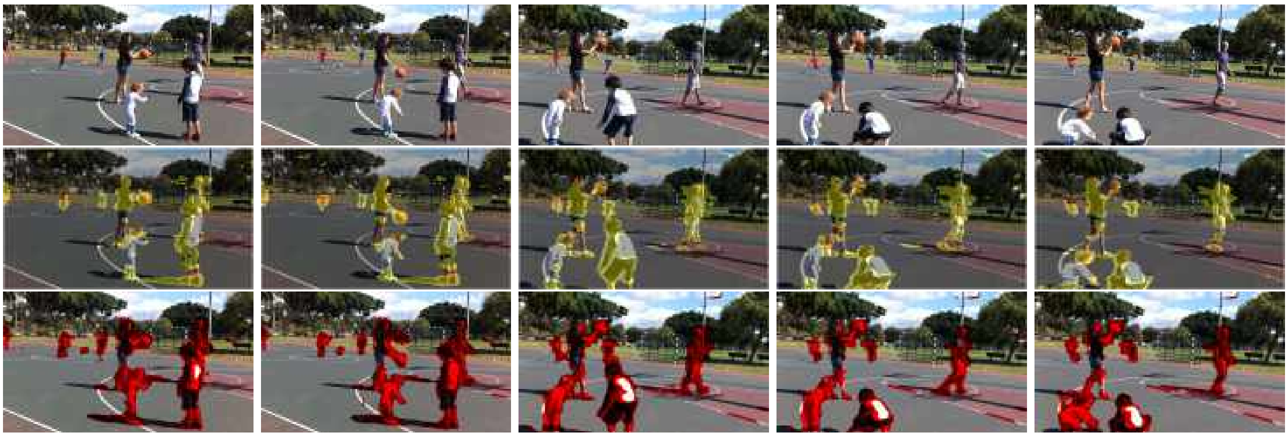

Fig. 4 The figure shows the comparison on the basketball dataset. The first row shows the input set of images. The second and third row show the results obtained in [10] and using our approach, respectively.

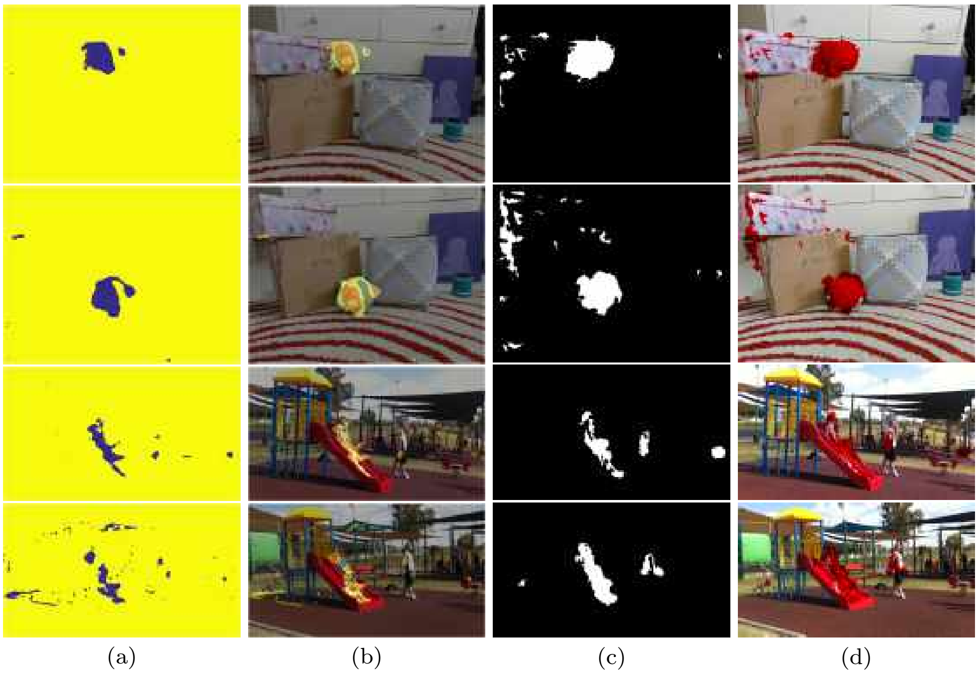

Fig. 5 The figure shows the comparison on 2 images of toy ball and playground dataset each. The first two rows show the comparison on the toy ball dataset while the third and the fourth row show the comparison on the playground dataset. The masks and the corresponding highlighted region shown in (a) and (b) are obtained in [10] while the masks and the corresponding highlighted region shown in (c) and (d) respectively are obtained using our approach.

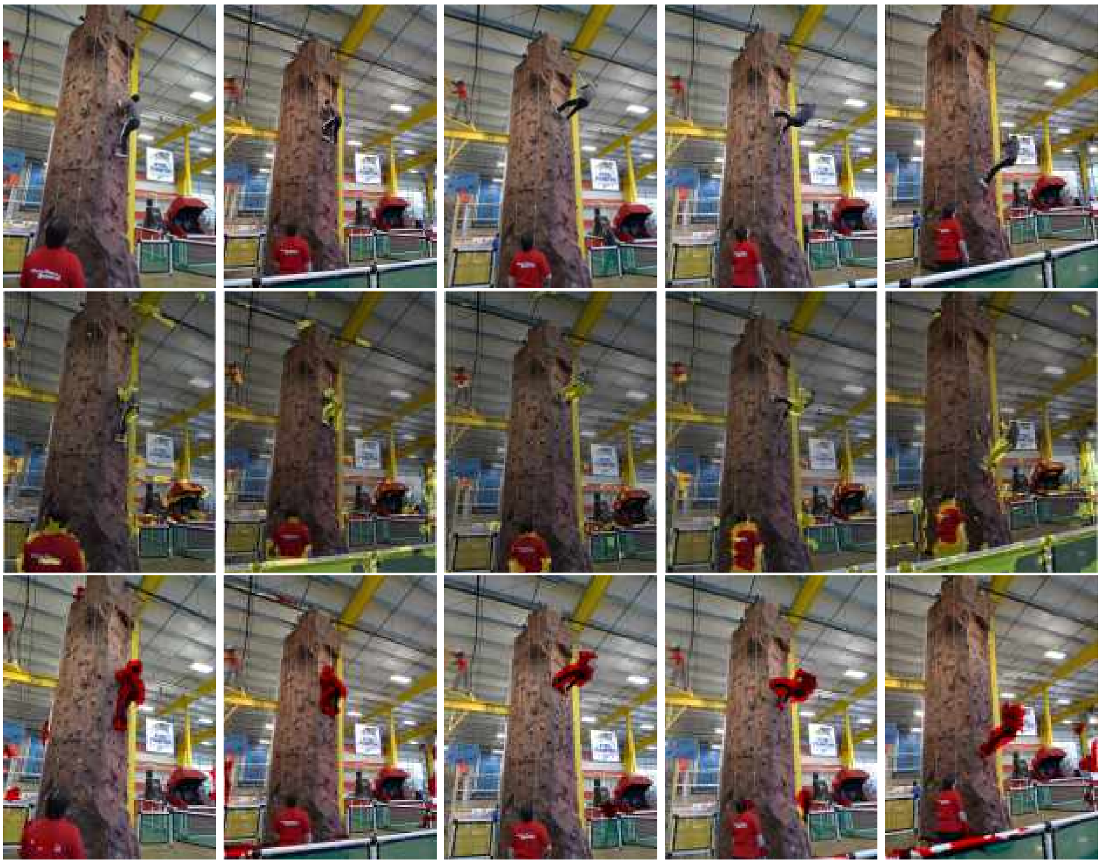

Fig. 6 The figure shows the comparison on the climbing dataset. The first row shows the input image stack. The second and the third row show the results obtained in [10] and our approach respectively.

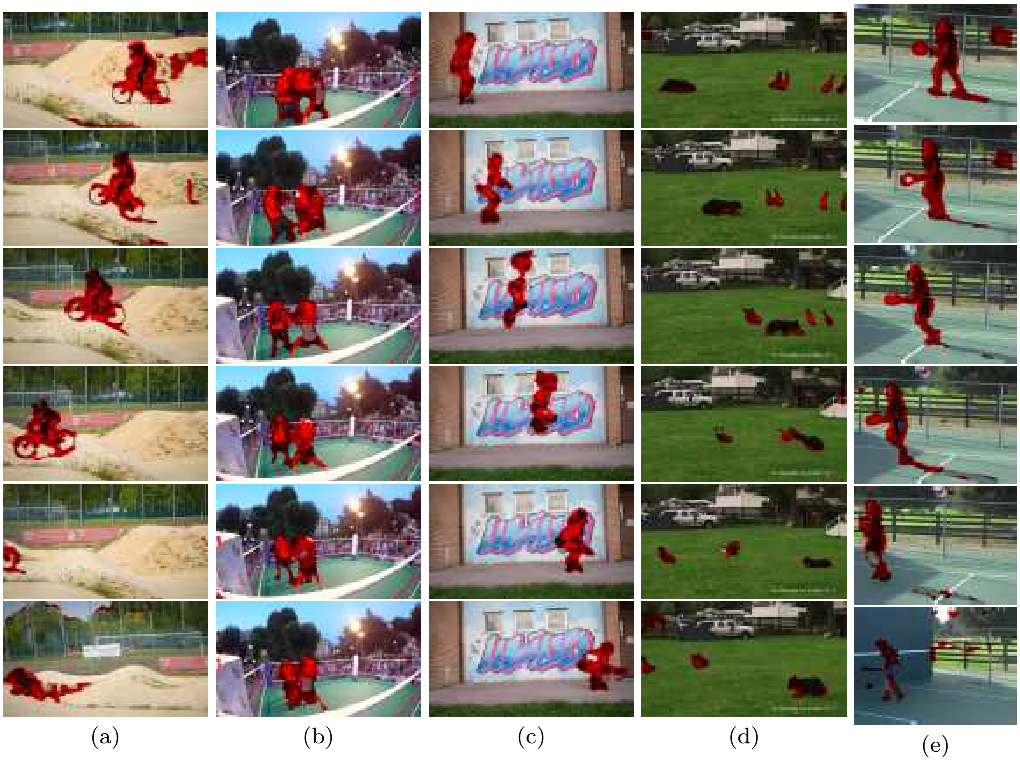

Fig. 7 The figure shows the results obtained on (a) bmx-bumps, (b) boxing-fixeye, (c) rollerblade, (d) dog-gooses and (e) tennis dataset using our approach. The results are obtained by making use of both the coherency and the epipolar constraint.

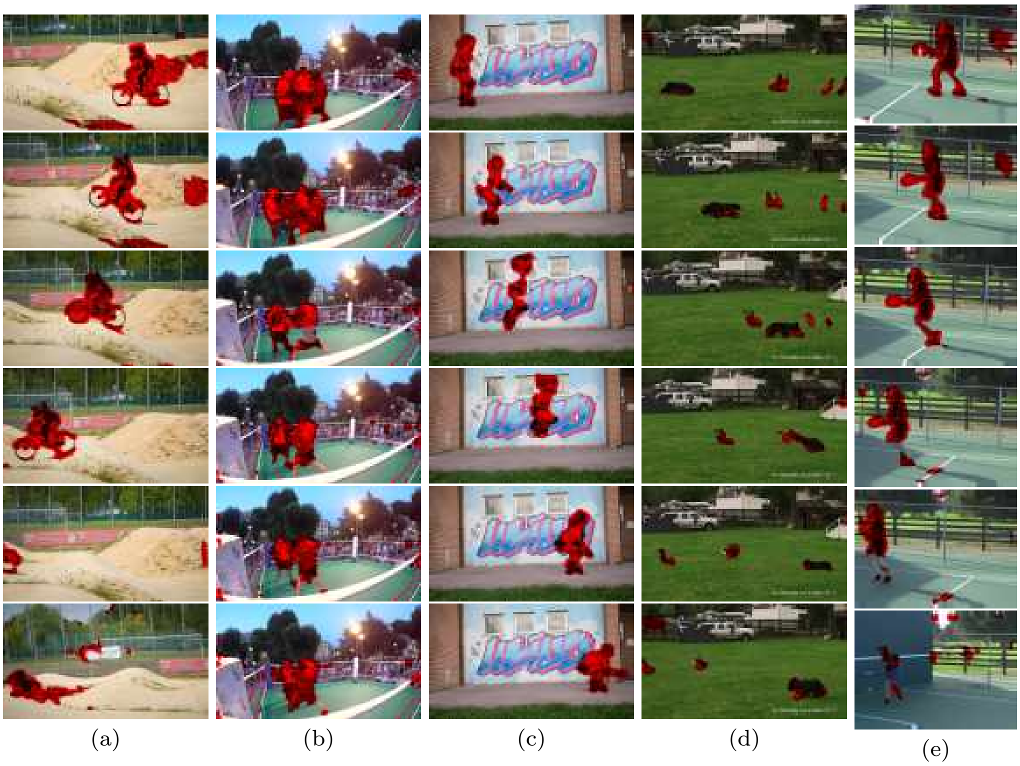

Fig. 8 The figure shows the results obtained on (a) bmx-bumps, (b) boxing-fixeye, (c) rollerblade, (d) dog-gooses and (e) tennis dataset using our approach. These results are obtained purely by exploiting the coherency present in the scenes i.e., $w(\boldsymbol{x}_r) = 0$ for all values of $\boldsymbol{x}_r$ (Eq. 20).

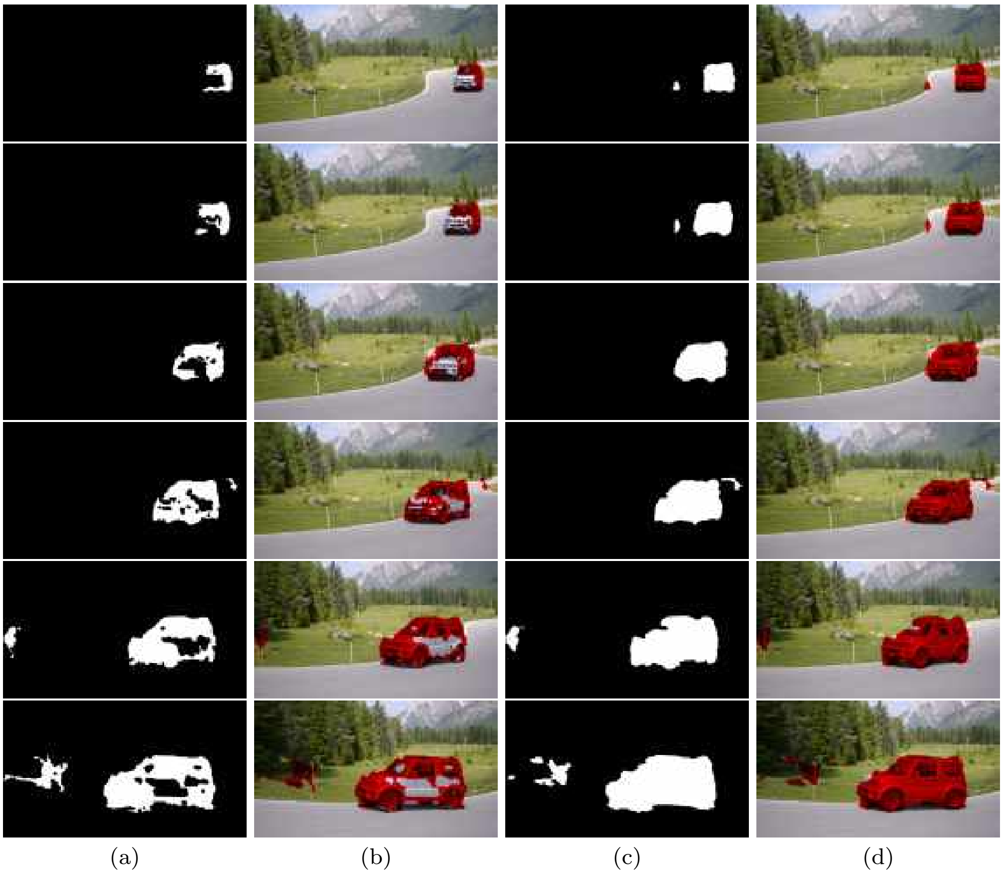

Fig. 9 The figure shows the result on car-turn dataset. (a) and (b) show the mask and the corresponding highlighted region obtained using the proposed approach considering both the coherency and the epipolar geometry. (c) and (d) show the mask and the corresponding highlighted region obtained using the proposed approach while relying only on the coherency present in the scene i.e., $w(\boldsymbol{x}_r) = 0$ for all values of $\boldsymbol{x}_r$ (Eq. 20).

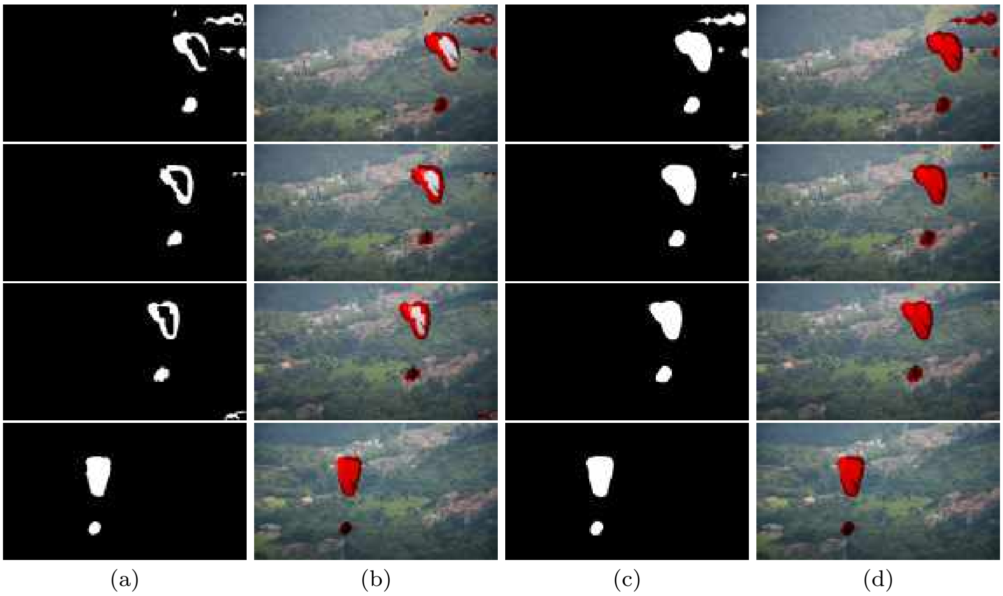

Fig. 10 The figure shows the result on paragliding dataset. (a) and (b) show the mask and the corresponding highlighted region obtained using the proposed approach considering both the coherency and the epipolar geometry. (c) and (d) show the mask and the corresponding highlighted region obtained using the proposed approach while relying only on the coherency present in the scene i.e., $w(\boldsymbol{x}_r) = 0$ for all values of $\boldsymbol{x}_r$ (Eq. 20).

by making use of both the coherency and the epipolar geometry. In Fig. 8, the results are obtained purely by exploiting the coherency present in the scenes i.e., $w(\boldsymbol{x}_r) = 0$ for all values of $\boldsymbol{x}_r$. Here, $\boldsymbol{x}_r$ refers to the locations in the reference image (Eq. 20). It can be seen that even without using any geometric information, we are able to segregate dynamic objects and the static region. In Fig. 9 and Fig. 10, we show that the obtained results are better in the case where we have only relied on the coherency present in the scene. In Fig. 9 and Fig. 10, (a) and (b) show the mask and the corresponding highlighted region obtained using the proposed approach on the car-turn and paragliding datasets. These results are obtained while considering the coherency and the epipolar geometry. In Fig. 9 and Fig. 10, (c) and (d) show the mask and the corresponding highlighted region obtained using the proposed approach while relying only on the coherency present in the scene. This could have happened due to the drawbacks of using epipolar geometry discussed in the Section 2 and the domination of the geometric term over the coherency term.

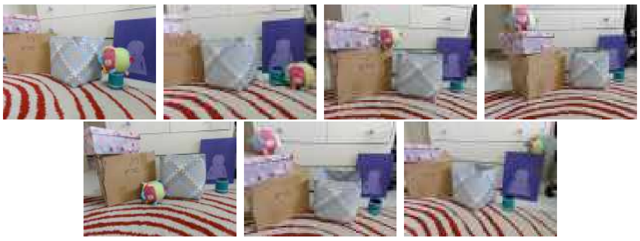

Fig. 11 Images from Toy ball dataset.

4.3 Limitations

The values of σ_1, σ_2, th, and the threshold used in DBSCAN algorithm are obtained empirically through experiments and these values are sensitive to the choice of feature descriptors. The objects with hardly any motion throughout the image sequence are labeled as static objects. This could be addressed using the semantic information about the scene. However, we have prevented ourselves from using any semantic information in this work. It can seen in Table 1, that the proposed algorithm does not perform better in the case of toy ball dataset. The scene in the toy ball dataset contains lots of texture and the camera is quite near to the objects which lead to the significant change in appearance during the camera motion. Hence, even the corresponding patches has significant difference among their features. We will address these limitations in the future work.

5 Conclusion and Future Work

We have developed a novel approach to detect dynamic objects present in the CrowdCam images. We exploit the coherency present in the scene along with the epipolar geometry between the pair of images. The proposed approach can handle complex scenes. This can be seen in the results obtained using our approach. The proposed approach does not require tracking a dynamic object or finding accurate matches over dynamic objects. It relies more on the matches produced in the static regions which are easier to obtain.
The updated nearest neighbor fields $\mathcal{N}^h_{r\to s}, \forall s = 1, \ldots, k$ and $h \in \{\downarrow, \uparrow\}$, contains the information regarding the corresponding location of the occluded static region in the source images which can be used to remove dynamic objects. Fig. 12 shows the removal of a dynamic object from a scene captured using a static camera. In the case of a static camera, we rely only on the coherency i.e., $w(\boldsymbol{x}_r) = 0$ for all values of $\boldsymbol{x}_r$, where $\boldsymbol{x}_r$ refers to the locations in the reference image (Eq. 20). We replace the pixel value at each location of the dynamic region (shown in the second row of Fig. 12) with the pixel value of its most confident nearest neighbor location. This neighbor location is obtained using the updated nearest neighbor maps and the confidence maps. However, due to camera motion, it is not an easy task in case of handheld cameras. We would like to address this problem in our future work by decoupling the camera motion and the motion of different objects with the help of the proposed algorithm. We further want to nullify the perspective deformation caused due to the camera motion on the source images with respect to the reference image. Following which, we apply the proposed algorithm and update the pixel values of the dynamic region as done in Fig. 12. We would also like to explore the possibility of utilizing super-pixels [1] or object-level segmentation [19, 42] in order to reduce the computational complexity and further improve the quality of the results [39, 36, 35].

References

1. Achanta R, Shaji A, Smith K, Lucchi A, Fua P, Süsstrunk S (2012) Slic superpixels compared to state-of-the-art superpixel methods. IEEE transactions on pattern analysis and machine intelligence 34(11):2274–2282
2. Agarwal S, Furukawa Y, Snavely N, Simon I, Curless B, Seitz SM, Szeliski R (2011) Building rome in a day. Communications of the ACM 54(10):105–112
3. Babenko B, Yang MH, Belongie S (2011) Robust object tracking with online multiple instance learn-

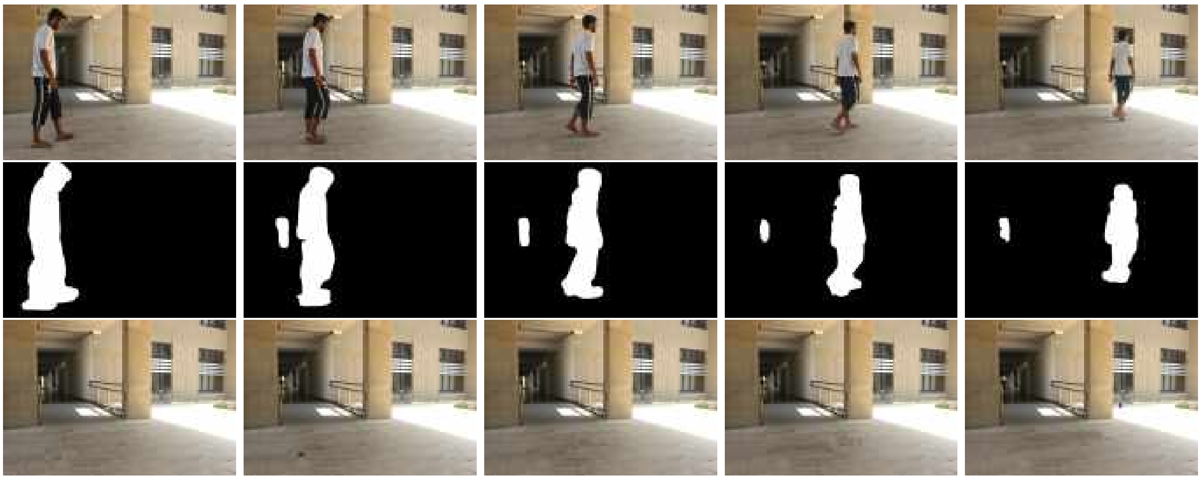

Fig. 12 The figure shows the detection and removal of dynamic object in a scene captured using static camera using the proposed approach. The first row shows the input image stack. The second row shows the object binary mask segregating the dynamic and the static region of the scene. The third row shows the images obtained after the removal of the dynamic object in the corresponding input images.

ing. IEEE transactions on pattern analysis and machine intelligence 33(8):1619–1632

4. Barnes C, Shechtman E, Finkelstein A, Goldman D (2009) Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics-TOG 28(3):24
5. Barnes C, Shechtman E, Goldman DB, Finkelstein A (2010) The generalized patchmatch correspondence algorithm. In: European Conference on Computer Vision, Springer, pp 29–43
6. Basha T, Moses Y, Avidan S (2012) Photo sequencing. European Conference on Computer Vision pp 654–667
7. Brox T, Malik J (2011) Large displacement optical flow: descriptor matching in variational motion estimation. IEEE transactions on pattern analysis and machine intelligence 33(3):500–513
8. Comaniciu D, Ramesh V, Meer P (2003) Kernel-based object tracking. IEEE Transactions on pattern analysis and machine intelligence 25(5):564–577
9. Cremers D, Soatto S (2005) Motion competition: A variational approach to piecewise parametric motion segmentation. International Journal of Computer Vision 62(3):249–265
10. Dafni A, Moses Y, Avidan S, Dekel T (2017) Detecting moving regions in crowdcam images. Computer Vision and Image Understanding 160:36–44
11. Dar M, Moses Y (2016) Temporal epipolar regions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 1220–1228
12. Dollár P, Zitnick CL (2013) Structured forests for fast edge detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1841–1848
13. Dosovitskiy A, Fischer P, Ilg E, Hausser P, Hazirbas C, Golkov V, van der Smagt P, Cremers D, Brox T (2015) Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp 2758–2766
14. Ester M, Kriegel HP, Sander J, Xu X, et al (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd, vol 96, pp 226–231
15. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6):381–395
16. Gullapally SC, Malireddi SR, Raman S (2015) Dynamic object localization using hand-held cameras. In: Communications (NCC), 2015 Twenty First National Conference on, IEEE, pp 1–6
17. HaCohen Y, Shechtman E, Goldman DB, Lischinski D (2011) Non-rigid dense correspondence with applications for image enhancement. ACM transactions on graphics (TOG) 30(4):70
18. Hartley R, Zisserman A (2003) Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press, New York, NY, USA
19. He K, Gkioxari G, Dollr P, Girshick R (2017) Mask r-cnn. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp 2980–2988
20. Korman S, Avidan S (2011) Coherency sensitive hashing. In: Proceedings of the IEEE International Conference on Computer Vision, IEEE, pp 1607–1614

21. Liu C, Yuen J, Torralba A (2011) Sift flow: Dense correspondence across scenes and its applications. IEEE transactions on pattern analysis and machine intelligence 33(5):978–994
22. Liu X, Cheung Ym, Tang YY (2016) Lip event detection using oriented histograms of regional optical flow and low rank affinity pursuit. Computer Vision and Image Understanding 148:153–163
23. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. International journal of computer vision 60(2):91–110
24. Margolin R, Zelnik-Manor L, Tal A (2014) Otc: A novel local descriptor for scene classification. In: European Conference on Computer Vision, Springer, pp 377–391
25. Moses Y, Avidan S, et al (2013) Space-time tradeoffs in photo sequencing. In: Proceedings of the IEEE International Conference on Computer Vision, pp 977–984
26. Ni Q, Wang J, Gu X (2015) Moving target tracking based on pulse coupled neural network and optical flow. In: International Conference on Neural Information Processing, Springer, pp 17–25
27. Ochs P, Malik J, Brox T (2014) Segmentation of moving objects by long term video analysis. IEEE transactions on pattern analysis and machine intelligence 36(6):1187–1200
28. Park HS, Shiratori T, Matthews I, Sheikh Y (2010) 3d reconstruction of a moving point from a series of 2d projections. In: European conference on computer vision, Springer, pp 158–171
29. Peng Y, Chen Z, Wu QJ, Liu C (2017) Traffic flow detection and statistics via improved optical flow and connected region analysis. Signal, Image and Video Processing pp 1–7
30. Perazzi F, Pont-Tuset J, McWilliams B, Gool LV, Gross M, Sorkine-Hornung A (2016) A benchmark dataset and evaluation methodology for video object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 724–732
31. Pont-Tuset J, Perazzi F, Caelles S, Arbeláez P, Sorkine-Hornung A, Van Gool L (2017) The 2017 davis challenge on video object segmentation. arXiv:170400675
32. Sevilla-Lara L, Sun D, Jampani V, Black MJ (2016) Optical flow with semantic segmentation and localized layers. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3889–3898
33. Shi J, Malik J (1998) Motion segmentation and tracking using normalized cuts. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1154–1160
34. Song Hj, Shen Ml (2011) Target tracking algorithm based on optical flow method using corner detection. Multimedia Tools and Applications 52(1):121–131
35. Tian L, Li M, Zhang G, Zhao J, Chen YQ (2017) Robust human detection with super-pixel segmentation and random ferns classification using rgb-d camera. In: Proceedings of the IEEE International Conference on Multimedia and Expo, IEEE, pp 1542–1547
36. Tian Z, Liu L, Zhang Z, Fei B (2016) Superpixel-based segmentation for 3d prostate mr images. IEEE transactions on medical imaging 35(3):791–801
37. Tola E, Lepetit V, Fua P (2010) Daisy: An efficient dense descriptor applied to wide-baseline stereo. IEEE transactions on pattern analysis and machine intelligence 32(5):815–830
38. Vedaldi A, Fulkerson B (2010) Vlfeat: An open and portable library of computer vision algorithms. In: Proceedings of the 18th ACM international conference on Multimedia, ACM, pp 1469–1472
39. Wang C, Chan SC, Zhu ZY, Zhang L, Shum HY (2016) Superpixel-based color–depth restoration and dynamic environment modeling for kinect-assisted image-based rendering systems. The Visual Computer pp 1–15
40. Wang JY, Adelson EH (1994) Representing moving images with layers. IEEE Transactions on Image Processing 3(5):625–638
41. Wang TY, Kohli P, Mitra NJ (2015) Dynamic sfm: Detecting scene changes from image pairs. In: Computer Graphics Forum, Wiley Online Library, vol 34, pp 177–189
42. Wei XS, Xie CW, Wu J, Shen C (2018) Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization. Pattern Recognition 76:704 – 714
43. Weinzaepfel P, Revaud J, Harchaoui Z, Schmid C (2013) Deepflow: Large displacement optical flow with deep matching. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1385–1392
44. Wu Y, Lim J, Yang MH (2013) Online object tracking: A benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 2411–2418
45. Zhang C, Chen Z, Wang M, Li M, Jiang S (2017) Robust non-local tv- l^1 optical flow estimation with occlusion detection. IEEE Transactions on Image Processing 26(8):4055–4067

46. Zhang G, Liu J, Li H, Chen YQ, Davis LS (2017) Joint human detection and head pose estimation via multistream networks for rgb-d videos. IEEE Signal Processing Letters 24(11):1666–1670
47. Zhu G, Porikli F, Li H (2016) Robust visual tracking with deep convolutional neural network based object proposals on pets. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp 26–33
48. Zontak M, Irani M (2011) Internal statistics of a single natural image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp 977–984