CrowdCam: Dynamic Region Segmentation

Nir Zarrabi  
Tel-Aviv University, Israel  
nirzl@mail.tau.ac.il

Shai Avidan  
Tel-Aviv University, Israel  
avidan@eng.tau.ac.il

Yael Moses  
The Interdisciplinary Center, Israel  
yael@idc.ac.il

Abstract

We consider the problem of segmenting dynamic regions in CrowdCam images, where a dynamic region is the projection of a moving 3D object on the image plane. Quite often, these regions are the most interesting parts of an image. CrowdCam images is a set of images of the same dynamic event, captured by a group of non-collaborating users. Almost every event of interest today is captured this way. This new type of images raises the need to develop new algorithms tailored specifically for it. We propose an algorithm that segments the dynamic regions in CrowdCam images. The proposed algorithm combines cues that are based on geometry, appearance and proximity. First, geometric reasoning is used to produce rough score maps that determine, for every pixel, how likely it is to be the projection of a static or dynamic scene point. These maps are noisy because CrowdCam images are usually few and far apart both in space and in time. Then, we use similarity in appearance space and proximity in the image plane to encourage neighboring pixels to be labeled similarly as either static or dynamic. We define an objective function that combines all the cues and solves it using an MRF solver. The proposed method was tested on publicly available CrowdCam datasets, as well as a new and challenging dataset we collected. Our results are better than the current state-of-the-art.

1. Introduction

CrowdCam images are a collection of images that capture a dynamic event. Such data is captured nowadays at almost every event of interest. The images are captured by different people with little or no coordination or synchronization (see examples in Fig. 1). We wish to detect and segment the dynamic regions in each of the images, independent of their class and shape. Detecting dynamic regions helps determine where the action is and highlight changes in a scene. As a by-product, discarding the dynamic regions can help algorithms that assume a static scene, such as Structure-From-Motion. The main challenge in detecting dynamic regions is to determine whether the image regions vary due to motion or due to CrowdCam characteristics (i.e. occlusions, illumination variance, viewpoint change, etc.).

The problem of detecting dynamic regions in CrowdCam images was first addressed by Dafni et al. [8]. Their method produces a dynamic score map for each image. The dynamic score of each pixel reflects whether the pixel is a projection of a dynamic or static point in the scene. The score is computed independently for each pixel in each of the images, using only geometric reasoning (see Sec. 3.1). Given the dynamic score map, a dynamic region segmentation can then be obtained by a naïve algorithm that applies a threshold to each of the score maps.

Our goal is to improve the dynamic score maps of [8] by adding cues on top of the geometric reasoning. Our cues are based on similarity in appearance between patches and proximity between pixels in the image plane. These cues capture short and long range relationship between pixels. Our key observation is that similar patches are likely to belong to the same class of objects, and objects that belong to the same class (e.g., grass, wall, or a person) are either static or dynamic.

We use Markov Random Field (MRF) to combine all the different sources of information (geometry, proximity, and appearance) to update the dynamic score for each pixel. For
each set of pixels from patches with similar appearance, we define an appearance distribution of the candidate scores based on other pixels in the set. The unary term of each pixel is a mixture of the geometry and appearance distributions. The binary term helps enforce the spatial smoothness assumption. This way, the probability distribution function of the unary term captures long-range pixel interactions and the MRF solver mitigates local errors and adjusts pixel labeling to comply with strong edges in the image. The results of the MRF is an updated dynamic score map for each image. To obtain the final segmentation result, a threshold is applied to the computed maps. We refer to our algorithm as CrowdCam Dynamic Region Segmentation, or CDRS.

The dynamic region segmentation task we address differs from semantic segmentation, co-segmentation, background subtraction, and motion segmentation in video sequences. Semantic segmentation, e.g. segmentation of a car or a person, does not indicate whether the object is dynamic or static. For example, a car may move or park. Co-segmentation methods do not assume that objects are dynamic, and motion segmentation in video relies on the proximity in space and time of the video frames. Background subtraction methods are also not applicable: CrowdCam images cannot be aligned by homography and the background cannot be learned.

We evaluate our method on previously published datasets, as well as a new and challenging dataset collected by us. Experiments show that our method outperforms existing state-of-the-art techniques. The main contributions of our paper are: (i) Using patches with similar appearance in a set of images of the same scene for improving dynamic region segmentation; (ii) Effectively combining the three cues: geometry, proximity and appearance; (iii) New challenging image sets\(^1\) for evaluating dynamic region segmentation, which can be used for further research of CrowdCam data.

2. Background

Detecting moving regions in CrowdCam images was addressed only in \cite{8, 10}. Both methods output dynamic score maps, using the observation that matching projections of static regions in the scene must be consistent with epipolar geometry. Our goal is to improve these score maps using proximity as well as appearance cues using patches with a similar appearance in the image itself and in other images of the scene.

Co-segmentation of objects, using a set of images, was first introduced by Rother et al. \cite{18}. Their method, however, does not use motion as a cue. All existing co-segmentation methods use single image cues for segmentation (e.g., color or texture), but they differ in the assumptions they make about the other images in the set. For example, a method might assume that the background differs between images (e.g., \cite{18}), that the foreground objects are salient in all images (e.g., \cite{19}), or that segmented regions are consistent (e.g., \cite{3}). None of the above-mentioned methods directly address moving region segmentation, as we do in our approach. Nor do we assume that the number of moving regions is known and that the moving objects are necessarily the salient objects in the scene. Note that in CrowdCam images the moving objects have the same background.

Using a set of images can also be used for extracting depth information, which can then be used as an additional source of information for object segmentation (e.g. \cite{9, 14, 20}). Such methods are prone to errors in depth estimation that, in turn, might affect segmentation results. Dafni et al. \cite{8} showed that existing 3D reconstruction methods fail on CrowdCam images because the images are few and far apart. See discussion in \cite{8}.

Semi-supervised methods use scribbles of foreground or background, provided by the user, for single image segmentation (e.g. \cite{6, 17, 24}) or co-segmentation (e.g. \cite{3, 7, 18}). An alternative to user scribbles is to use an exact segmentation in one image and propagate it to the rest of the CrowdCam images \cite{1}. The geometric scores we use can also be regarded as an external input for a segmentation or co-segmentation method. However, the user scribbles are sparse and accurate, whereas our score maps are dense and very noisy.

We use MRF as a basic optimization solver for our task. MRF was used in many single image segmentation methods (e.g., \cite{17}) and in co-segmentation methods (e.g., \cite{3}). These methods differ mainly in their definitions of the unary and the binary terms. We define a new unary term, that incorporates scores of patches with a similar appearance from all images. Moreover, our method uses MRF to compute a dynamic score map rather than the final segmentation. Most similar to our approach of using proximity and appearance, is the fully-connected Conditional Random Field (CRF) \cite{13}. We discuss this method and compare it to our results in Sec. 4.1.

3. Method

The input to our method is a set \( \mathcal{I} \) of still CrowdCam images of a dynamic scene, taken by a set of uncalibrated and unsynchronized cameras from a wide baseline setup. We use a modification of the method of Dafni et al. \cite{8} to compute the initial score map of each image that reflects whether a pixel is a projection of a static or a dynamic 3D point. Equipped with this initial guess, we propose an MRF solution. Our solution integrates local information and information across multiple images to obtain better score maps, resulting in a far superior result. The

\(^1\)Our dataset will soon be publicly available
outline of the method is depicted in Fig. 2.

3.1. Initial Map Calculation

We give a short description of the method of Dafni et al. [8] for completeness. They assume that corresponding static pixels from multiple images must satisfy the epipolar geometry constraints. This observation is used to compute a per-pixel score that encodes whether the pixel is static. The epipolar geometry (i.e., the fundamental matrix) is computed for a reference image \( I \in \mathcal{I} \), and each image \( I' \in \mathcal{I} \) using computed matching features and the RANSAC method. An initial static score of a pixel is calculated based on whether there exists at least one potential match along the epipolar lines, which is not necessarily the correct one. Pixels might not have a match since they are a projection of a moving 3D point, but also due to occlusions, variations in camera parameters, out of the field of view, etc. Hence, multiple score maps are calculated, with respect to the other images in \( \mathcal{I} \), and the dynamic score map is obtained by their combination. For more details, see [8].

The output of this method is an initial score map \( s_r \) = \{s_1, \cdots, s_n\} for each image \( r \), where \( n \) is the number of pixels in an image. The score map is computed using the geometric consideration; hence, we regard it as a geometric score. The score value \( s_j \in [0, 1] \) encodes whether pixel \( j \) is a projection of a static (i.e. \( s_j = 0 \)) or dynamic (\( s_j = 1 \)) 3D scene point.

In our implementation of this method, we calculated the fundamental matrices using DeepMatching [23] to compute sparse matches between images. To determine patch similarity, we used deep features in a similar fashion to [11] and [22] rather than the HoG features used in [8]. The initial feature map for each image was generated by concatenating original results of [8] (see Sec. 4).

3.2. MRF Solution

Given a set of maps \( \{s_r\} \) as described above, we use a Markov Random Field (MRF) approach to compute a new score map for each image. We use a finite number of labels to represent the score values (the initial scores are real numbers). Thus, the space of scores becomes discrete, and each pixel will be assigned at the end of our method to one of the score labels \( \mathcal{L} = \{l_1, \cdots, l_b\} \) (we use \( b = 30 \)).

Let \( X = \{X_1, \cdots, X_n\} \) denote a random field of \( n \) variables (\( n \) is the number of pixels in an image), and let the domain of each variable be the set of score labels \( \mathcal{L} (|\mathcal{L}| = b) \). Our goal is to compute for each \( X_j \) a single label \( x_j \in \mathcal{L} \). That is, we wish to find a labeling \( x \in \mathcal{L}^n \) that minimizes the standard MRF energy objective function:

\[
E(x) = \sum_{j=1}^{n} \Psi_u(x_j) + \lambda \sum_{j=1}^{n} \sum_{i \in \mathcal{N}(j)} \Psi_p(x_j, x_i),
\]

where \( \mathcal{N}(j) \) is the neighborhood of pixel \( j \), \( \Psi_u(x_j) \) is the unary potential, and \( \Psi_p(x_j, x_i) \) is the pairwise potential. The question is how to compute the unary potential \( \Psi_u(x_j) \).
We set $\Psi_u(x_j)$ to be the function of a mixture of two distributions: $h_j^G$, which is a discretization of pixel $j$’s initial score, $s_j$, and $h_j^A$, which is based on the score of pixels in the image set that have similar feature vectors to pixel $j$. We use the letter $G$ in our denotation of $h_j^G$ because this term relies on geometry. It is based on all pixels along the epipolar lines associated with pixel $j$ (described in Sec. 3.1). We use the letter $A$ in our denotation of $h_j^A$ because this term relies on appearance. It is based on all pixels that are similar in appearance to pixel $j$.

The distribution $h_j^G$ of labels for pixel $j$ is a delta function of the discretized initial score $s_j$ of that pixel:

$$h_j^G(l_m) = p(x_j = l_m) = \begin{cases} 1 & \text{if } m = \lceil s_j \cdot b \rceil \\ 0 & \text{otherwise} \end{cases}. \quad (2)$$

The weight $w_i$ is based on the Euclidean distance in feature space between $f_i$ and the mean feature vector of all pixels

$$w_i = e^{-\beta \frac{d_i^2}{2\sigma^2}}. \quad (4)$$

The weight $w_i$ is based on the Euclidean distance in feature space between $f_i$ and the mean feature vector of all pixels
in the cluster. $M$ is the median of the distances of all features in the cluster to the cluster center. We set $\beta = 0.3$ in all our experiments. This function is chosen such that all features affect the cluster pdf while features far from the cluster center contribute less. Note that the pixels assigned to a cluster can come from different images and are not limited to a single image. Combining the two distributions, we have,

$$ h_j = \alpha h_j^G + (1 - \alpha) h_j^A. \quad (5) $$

Now, we set the unary term to be:

$$ \Psi_u(x_j = l_m) = -\log h_j(l_m). \quad (6) $$

The parameter $\alpha \in [0, 1]$ is the mixture weight. The effect of $\alpha$ is shown in Fig. 6(a) and discussed in Sec. 4.3.

An example of the computation of the probability distribution function is presented in Fig. 4. It shows two different images from the same set with pixels $i$ and $j$ marked on them. Both pixels belong to the same cluster, and both depict a static 3D scene point of the same surface, so their score should be low. In practice, however, their scores are quite high (one has a score above 0.4 and the other has a score above 0.8). The appearance-based distribution of their cluster indicates that most pixels in that cluster are more likely to be static (i.e., have a low score). The unary term of each pixel is taken to be a mixture of both geometry-based and appearance-based pixel statistics. The final result of the MRF shows that the score of the top pixel drops dramatically from over 0.8 to about 0.2, while the score of the bottom pixel does not change much. This shows the importance of using both appearance and proximity.

The output of the MRF is a new set of score maps for each image $\hat{s}_r$ that significantly improves the dynamic object segmentation over the initial map $s_r$, as described in the next section.

As for the implementation, to solve the MRF energy minimization equation we used the graph-cut solver published by [4, 5, 12].

4. Experiments

We implemented our method using MATLAB. We tested our method on the dataset used by [8] as well as on a new dataset we captured. The results of our method are compared to the results of the only two methods that address the task of dynamic object segmentation from CrowdCam data, Dafni et al. [8] and Kanojia et al. [10]. We also compare our results with ML estimator (see Sec. 4.2), and with Dense-CRF [13]. In addition, we evaluated the effect of various parameters of our method on the obtained results.

Data Sets: The images in each set were captured by several cameras, at a different timing from different viewpoints, as typical for CrowdCam images. The dataset used by [8] contains six scenes, with one or more dynamic objects such as people, bicycles and toy balls. The images were taken by Dafni et al. [8], Park et al. [16], and Basha et al. [2]. Each set consists of four to ten images. We collected a new dataset to extend the variety of considered scenes. Our new dataset contains 14 scenes, with dynamic objects such as people and animals. Each set consists between five to ten images. Examples of scenes are presented in Fig 5, and in the supplementary material. We used 20 scenes in our experiments.

Evaluation: The segmented ground truth masks of dynamic objects were provided by [8] and for the new dataset, they were manually marked. In [8] a few scenes (Climbing,
Skateboard and Playground) are marked for both ‘ground truth’ (of the dynamic objects) and ‘don’t care’ regions, which are disregarded when evaluating the results. The Jaccard measure, also known as Intersection over Union (IoU), is a standard measure for binary map segmentation results. It is used to compare the computed and the ground-truth region.

The output of our algorithm, as well as the discussed algorithms (CRF, ML and Dafni et al. [8]), is a score map for each image rather than a binary map. Inspired by the evaluation methodology of the Berkeley Segmentation Dataset [15] and [8], we obtain a binary map by applying a threshold operation on the score maps and then use the Jaccard measure. We consider two thresholds for our evaluation. The first, per-image, is chosen independently for each image such that they maximize the Jaccard measure for each image. The second, per-set, is applied to all images in a given set, such that it maximizes the average Jaccard measure of all images in a set.

### 4.1. Results

Here we present the results of our method on all datasets described above using the same set of parameters for all image sets: \( \alpha = 0.2 \) when calculating the final dynamic score PDF \( h_j \) (Eq. 5), \( \lambda = 450 \) in the MRF energy objective function (Eq. 1), and \( K = 600 \) as the number of clusters when calculating \( h_j^A \) (Eq. 3). Two levels of sharing information were considered. The first is when \( h_j^A \) is computed from a single image (CDRSs), and the second is when it is computed from multiple images in the set (CDRSm).

Qualitative results are shown in Fig. 3 and in Fig. 5. Fig. 3 illustrates the strength of our method, as a much better segmentation is obtained using the calculated score maps. The score maps calculated by our method are more accurate thanks to our algorithm’s sharing of statistics between neighboring pixels in both appearance and spatial domains. Fig. 5 presents images from a few scenes that reveal the difficulties of dynamic region segmentation in CrowdCam image sets. The top two rows show that a good segmentation is obtained even when dynamic objects appear different for various reasons. This includes changes in viewpoint, differences in color, changes in illumination, and the use of different cameras. Multiple objects are segmented as our method gives no importance to the number of dynamic objects to segment. This is clearly seen in the segmentation of “Zoo4” where all monkeys are correctly segmented, while a small portion of the background is erroneously segmented as a moving region as well. The last row presents a very challenging scene of a moving bear (“Zoo8”). The results for this scene are not as good because the bear has a similar appearance in texture and color to its background. The initial score maps are very inaccurate as a result, and thus our method fails on this scene.

The quantitative results using the per-image and per-set thresholds are summarized in Table 1. The results of using multiple images, CDRSm, are better than those ob-
We compare our results with the results obtained by the only two algorithms designed to solve the same task considered in the paper, [8, 10]. It is apparent from several examples in Fig. 3 that the score maps produced by our method, as well as the final segmentation, improve over the initial score map computed by [8]. For the qualitative comparison, we present in Table 1 the reported results of the algorithm used in [8] and [10] for the dataset in [8], and we refer to these algorithms Dafni and Kanojia, respectively. We also modified the original implementation of [8], by using VGGs as features (instead of HOGs) (DafniVGG), as described in Sec. 3.1. Note that [10] reported only the per-set threshold results. The results show that using VGG improves the original results of [8] by 0.07 of the Jaccard measure. Our method outperforms the existing methods for CrowdCam dynamic object segmentation, [8, 10]. On average, we improved the results of [8] by more than 0.17 of the Jaccard measure after the change in features (DafniVGG) and outperformed the results of [10] by 0.16 of the Jaccard measure.

### Comparison to DenseCRF

Many available segmentation algorithms can be considered, given the score computed by [8] as input. We chose to compare our results to the DenseCRF [13] algorithm, that computes a label per pixel given a pdf of each pixel. The DenseCRF algorithm is somewhat similar to our algorithm in the sense that both search for the correct label per pixel, using the pixel’s neighbors in both spatial and feature spaces. The main difference is that the DenseCRF incorporates both spatial and feature neighborhood constraints in the pairwise potential of the energy minimization problem, while we incorporate the feature neighborhood constraint in the unary potential. Notice that we use deep-features when computing $h_j$ (as part of the unary term calculation) while [13] use RGB values in the energy minimization pairwise term. Another important difference is that we use information from all images in a set, whereas the DenseCRF considers pixels from the image itself while assigning greater importance to closer pixels.

In practice, the DenseCRF uses a pdf as input rather than a single score. Hence, we smoothed the score values by DafniVGG and used it as a pdf input. The results are summarized in Table 1. Using threshold on the score maps, we present better segmentation results, with an average Jaccard score difference of 0.07 in comparison to CDRSs and 0.1 in comparison to CDRSm.

| Scene (#images) | Dafni [8] | DafniVGG | Kanojia [10] | DCRF [13] | ML | CDRSs | CDRSm |
|----------------|-----------|----------|-------------|-----------|----|-------|-------|
| Helmet (4)     | 0.53 \ 0.36 | 0.69 \ 0.65 | -           | 0.69 \ 0.66 | 0.66 \ 0.62 | 0.77 \ 0.77 | 0.83 \ 0.83 |
| Climbing (10)  | 0.15 \ 0.13 | 0.40 \ 0.36 | - \ 0.34   | 0.42 \ 0.38 | 0.30 \ 0.28 | 0.59 \ 0.52 | 0.59 \ 0.59 |
| Skateboard (5) | 0.44 \ 0.42 | 0.44 \ 0.42 | - \ 0.50   | 0.49 \ 0.42 | 0.42 \ 0.39 | 0.71 \ 0.61 | 0.67 \ 0.67 |
| Toy-ball (7)   | 0.63 \ 0.60 | 0.60 \ 0.57 | - \ 0.44   | 0.66 \ 0.60 | 0.54 \ 0.52 | 0.74 \ 0.66 | 0.77 \ 0.64 |
| Playground (7) | 0.37 \ 0.32 | 0.40 \ 0.33 | - \ 0.37   | 0.41 \ 0.30 | 0.39 \ 0.35 | 0.50 \ 0.41 | 0.48 \ 0.48 |
| Basketball (8) | 0.48 \ 0.47 | 0.48 \ 0.46 | - \ 0.51   | 0.64 \ 0.60 | 0.48 \ 0.46 | 0.59 \ 0.54 | 0.63 \ 0.61 |
| Jim1 (8)       | -          | 0.57 \ 0.51 | -           | 0.61 \ 0.60 | 0.48 \ 0.44 | 0.69 \ 0.66 | 0.72 \ 0.70 |
| Jim2 (8)       | -          | 0.60 \ 0.56 | -           | 0.74 \ 0.64 | 0.59 \ 0.56 | 0.78 \ 0.74 | 0.84 \ 0.82 |
| Chess (5)      | -          | 0.24 \ 0.24 | -           | 0.34 \ 0.32 | 0.33 \ 0.32 | 0.33 \ 0.33 | 0.37 \ 0.36 |
| Shelf (8)      | -          | 0.44 \ 0.27 | -           | 0.53 \ 0.36 | 0.40 \ 0.28 | 0.61 \ 0.37 | 0.58 \ 0.48 |
| Zoo1 (7)       | -          | 0.31 \ 0.28 | -           | 0.32 \ 0.26 | 0.28 \ 0.24 | 0.37 \ 0.35 | 0.36 \ 0.35 |
| Zoo2 (8)       | -          | 0.12 \ 0.10 | -           | 0.11 \ 0.09 | 0.09 \ 0.07 | 0.24 \ 0.15 | 0.17 \ 0.11 |
| Zoo3 (6)       | -          | 0.29 \ 0.27 | -           | 0.47 \ 0.42 | 0.32 \ 0.30 | 0.60 \ 0.52 | 0.68 \ 0.63 |
| Zoo4 (10)      | -          | 0.46 \ 0.35 | -           | 0.70 \ 0.51 | 0.44 \ 0.33 | 0.60 \ 0.42 | 0.65 \ 0.52 |
| Zoo5 (7)       | -          | 0.33 \ 0.31 | -           | 0.40 \ 0.34 | 0.35 \ 0.33 | 0.45 \ 0.38 | 0.48 \ 0.46 |
| Zoo6 (9)       | -          | 0.13 \ 0.10 | -           | 0.25 \ 0.17 | 0.15 \ 0.09 | 0.29 \ 0.20 | 0.22 \ 0.20 |
| Zoo7 (9)       | -          | 0.17 \ 0.12 | -           | 0.17 \ 0.12 | 0.18 \ 0.11 | 0.27 \ 0.14 | 0.25 \ 0.18 |
| Zoo8 (8)       | -          | 0.16 \ 0.13 | -           | 0.17 \ 0.14 | 0.15 \ 0.12 | 0.33 \ 0.26 | 0.40 \ 0.32 |
| Zoo9 (6)       | -          | 0.36 \ 0.33 | -           | 0.47 \ 0.41 | 0.36 \ 0.32 | 0.61 \ 0.53 | 0.71 \ 0.67 |
| Zoo10 (7)      | -          | 0.29 \ 0.26 | -           | 0.46 \ 0.41 | 0.30 \ 0.26 | 0.45 \ 0.37 | 0.55 \ 0.43 |
| Mean           | -          | 0.37 \ 0.33 | -           | 0.43 \ 0.39 | 0.36 \ 0.32 | 0.53 \ 0.45 | 0.55 \ 0.50 |

Table 1. Results on CrowdCam datasets in terms of Jaccard index. The threshold on the score maps is chosen per image \ per set. ML: using only the unary term in the MRF equation. CDRSs: our algorithm using a single image when computing $h_j$. CDRSm: our algorithm using multiple images when computing $h_j$.
As for the implementation, using the code published by [13], we noticed that every change in DenseCRF parameters resulted in improving the results of some scenes while having the opposite effect on others, due to the significant differences in lighting and camera parameters. For that reason we performed grid search to find the DenseCRF parameters that achieve the best overall Jaccard results.

4.2. Algorithm Variants

We also tested the effect of using MRF on the computed pdf, by applying an ML estimator instead. That is, the final score of each pixel is the label with the maximal value of the pdf and proximity is not used. This is equivalent to setting $\lambda = 0$ in Eq. 1 (i.e., only the unary term is used without the pairwise term). The quantitative differences are shown in Fig. 3, and the qualitative ones in Table 1. It is apparent that the use of MRF is crucial for the performance of the method as it removes noise and enforces spatial smoothness. The results when using the ML estimator drop by 0.18 on average, in comparison to CDRSm.

4.3. Effect of Parameters

In this section we test the effect of our algorithm parameters, $\alpha$ (Fig. 6(a)), $\lambda$ (Fig. 6(b)), and $K$ of the $K$-means algorithm (Fig. 6(c)).

$\alpha$: The mixture weight from Eq. 5 determines the extent to which the geometric and appearance cues affect the pixels’ pdf. Setting $\alpha = 1$ causes the pixels’ pdf to depend solely on the geometry based score, ignoring global information. Setting $\alpha = 0$ has the opposite effect: each pixel loses its local initial score. It is clear from Fig. 6(a) that setting $\alpha$ closer to 1 yields poorer results. This implies that the global appearance pdf, $h^A_{j}$, is crucial for our method to work properly.

$\lambda$: The $\lambda$ parameter moderates the spatial constraint in our method. It weights the pairwise potential in the MRF energy term of Eq. 1 such that a higher $\lambda$ value results in a smoother scoring result. Fig. 6(b) illustrates the importance of the MRF solver and $\lambda$. When $\lambda$ is set to a small value, the spatial constraint has less effect on the final labeling, causing noisy results. This can also be seen in Fig. 3(c), where the ML estimator is obtained by setting $\lambda = 0$, i.e., using only the unary term. Setting a proper value for $\lambda$ significantly improves the results, as seen in Fig. 3(d). There is a wide range of $\lambda$ values where our algorithm works properly, as shown in Fig. 6(b).

$K$ in $K$-means: We use K-means clustering in our method (see Sec. 3 for details). The clustering is unsupervised, meaning the clusters are not partitioned to dynamic/static regions as they represent patches from all the images of the scene. In addition, we wish to sample as much appearance variety as possible, while keeping in mind the sharing of scores (i.e., Eq. 3). We tested the effect of $K$, the number of clusters in the K-means algorithm, on the final Jaccard result. The robustness of our method to $K$ is shown in Fig. 6(c). As expected, for a value greater than $\sim$200, the method performs well and the results do not change by much when increasing this value.

5. Conclusions

An approach to dynamic objects segmentation in CrowdCam images was presented. Our approach combines geometry, appearance and proximity. Geometric reasoning is used to produce a rough score value for each pixel in every image. This score determines how likely is a pixel to be the projection of a static or dynamic scene point. These noisy scores are further refined using proximity cue that encourage nearby pixels in the image plane to behave similarly, and appearance cue that encourage pixels with similar appearance to behave similarly. We tested our method on existing datasets, as well as a new dataset captured by us. Experiments suggest our method surpass the current state-of-the-art.

References

[1] H. Averbuch-Elor, J. Kopf, T. Hazan, and D. Cohen-Or. Co-segmentation for space-time co-located collections. The Vi-
2. T. Basha, Y. Moses, and S. Avidan. Photo sequencing. In European Conference on Computer Vision, pages 654–667. Springer, 2012.

3. D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen. iCoseg: Interactive co-segmentation with intelligent scribble guidance. In IEEE Conference on Computer Vision and Pattern Recognition, pages 3169–3176, June 2010.

4. Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Patt. Anal. Mach. Intell., 26(9):1124–1137, Sept 2004.

5. Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. IEEE Trans. Patt. Anal. Mach. Intell., 20:1222–1239, 2001.

6. Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In International Conference on Computer Vision, volume 1, pages 105–112. IEEE, 2001.

7. M. D. Collins, J. Xu, L. Grady, and V. Singh. Random walks based multi-image segmentation: Quasiconvexity results and gpu-based solutions. In IEEE Conference on Computer Vision and Pattern Recognition, pages 1656–1663. IEEE, 2012.

8. A. Dafni, Y. Moses, S. Avidan, and T. Dekel. Detecting moving regions in CrowdCam images. IEEE Trans. Patt. Analy. Image Understanding, 160:36–44, 2017.

9. S. Jeong, J. Lee, B. Kim, Y. Kim, and J. Noh. Object segmentation ensuring consistency across multi-viewpoint images. IEEE Trans. Patt. Analy. Mach. Intell., 40(10):2455–2468, 2018.

10. M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In IEEE Conference on Computer Vision and Pattern Recognition, pages 746–760. Springer, 2012.

11. K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

12. P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In International Conference on Computer Vision, Sydney, Australia, Dec. 2013.

13. H. Yu, Y. Zhou, H. Qian, M. Xian, and S. Wang. Loosecut: interactive image segmentation with loosely bounded boxes. In International Conference on Image Processing, pages 3335–3339. IEEE, 2017.