Counting in High Density Crowd Videos

Edgar Lopez
University of Texas at El Paso
ealopez12@miners.utep.edu

Haroon Idrees
University of Central Florida
haroon@cs.ucf.edu

Abstract

We propose a method for getting an estimate count of people in very high-dense crowd videos by extending a static crowd count method. Due to the challenging problem of perspective, occlusion, clutter, and low resolution counting by detection is not possible. Therefore, existing methods use dense features to get an estimate count. We propose using the extra information of motion in videos to construct a 3-D Conditional Random Field to add the constraint that nearby patches (superpixels) should have a similar count and in the time dimension patches connected by trajectories should also have similar count. The method was tested on a dataset of 13 videos and compared to our previous method that estimates counts in images. Results of our method show a 4% improvement from the static crowd count method.

1. Introduction

Crowds occur in a variety of situations including political speeches and rallies, concerts, marathons, stadiums, religious events, and others important events that happen throughout the world. Crowd analysis is an important task that helps in the management of crowds for safety and security. Having a crowd count often helps analyze the importance of events. It helps determine the need of development of public transportation infrastructure. Since counting the number of people by hand is inefficient, we need the help of Computer Vision methods to solve the various problems with crowd analysis, including counting of people.

There are many challenges in crowd counting such as occlusion, clutter, low resolution, and camera perspective. The best way to detect humans is by using face detection but in the case of high density crowds this is almost an impossible task. Therefore, other methods need to be used in order to have a crowd estimate count. Methods include using several sources in order to estimate a count such as low confidence head detections, texture elements, and frequency-domain analysis to estimate counts [1].

In our proposed method we start by computing SLIC superpixels [2]. Dense trajectories [3] are then extracted from the video. We use our previous method [1] to get a count per superpixel. A Conditional Random Field (CRF) is constructed to constraint counts in both the spatial and time dimension by using the count per superpixel and the dense trajectories. We minimize the costs of the CRF [4] to assign a new count to each superpixel to smooth counts across frames. Our proposed method was compared to using counts from our static image crowd count method [1] alone and results show a 4% improvement in our extension of the method to videos.

The organization of the paper is as follows. The proposed method was tested on the dataset described in section 2. Related work will be discussed in section 3. More details of the proposed framework will be discussed in section 4. The experiment details and results will be discussed in section 5, followed by a conclusion of the paper in section 6.

2. Dataset

Our method was tested on a dataset of 13 videos. The number of frames per video ranges from 50 to 900 frames. There is an average of 162 frames per video. Several frames in the videos were manually annotated with a dot for every head in the frame. There is an average of 9 annotated frames per video. Counts range from 208 to 1906. The average count per frame is 557. Example frames of the dataset are shown in Figure 1.

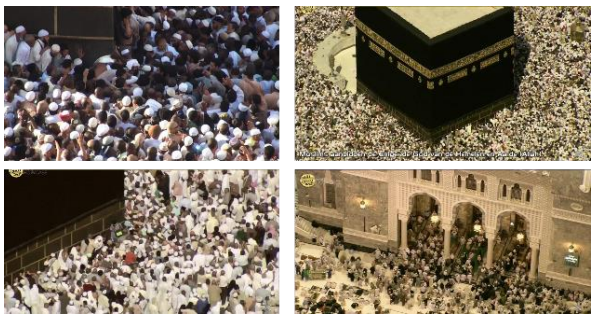


Figure 1: Example frames of our dataset. Crow density is very diverse throughout the dataset.

3. Related Work

Crowd analysis has become a topic of interest for the research community. Topics of interests within crowd analysis include having a count for high dense crowds in both images and videos. Existent crowd counting techniques address the problems in different ways. In some literature papers person detection is used for counting. Using person detection may be a suitable approach but only when there's few people in the scene. In the case where there are hundreds or thousands of people, counting by human detection is currently an impossible task. Person detection will not work in a dataset similar to the one described in section 2. In Figure 1 we can see the challenges in crowd counting such as occlusion, high density, appearance differences, low-resolution, and differences in camera perspective.

Current methods to count in high dense crowds include the use of density functions, to calculate a count, such as the ones employed in [5, 6]. Lempitsky and Zisserman [5] have a method to calculate counts in both microscopy cells and crowd scenes. They train a classifier to use SIFT descriptors to classify each pixel in a frame with a density count.

Rodriguez et al. [6] also use a density approach to calculate a crowd count. They use density functions to formulate person detection. They combine both density scores and human detection to generate a count. The use of head detection aids in the instances in which the perception in images changes, and head detection is possible in sections of the frames.

The proposed method is an extension of using a density approach to calculate counts in images. In [1] the proposed method uses density information as well as a head detection score to count in patches. Counting in patches makes the process homogenous. To generate the count for a whole image the counts in all of the patches are added. We proposed extending this method to count in high-dense crowd videos.

4. Framework

Given a crowd video our goal is to estimate a count of the number of people across frames. This method can be used to estimate a count for each frame in a video. Since crowd density does not change a lot across frames this method can be used more effective to calculate counts every other p frames, with p being any arbitrary integer. Figure 2 shows a flow chart of our method.

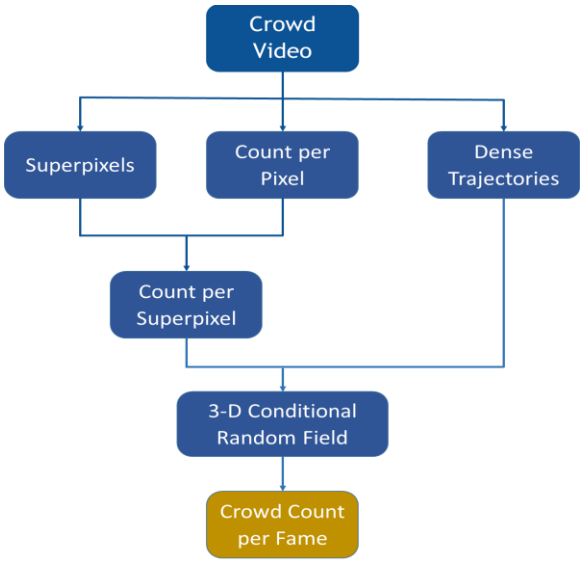


Figure 2: Flow chart of our method. The input to the method is a crowd video, with the output being crowd counts across frames.

4.1 SLIC Superpixels

Superpixels preserve appearance boundaries. They separate regions with different textures and shapes. We compute SLIC superpixels, using [2], across frames. For the experiments conducted, for the region size parameter we used the size of the length of the frames divided by 15. For the regularizer parameter we used region size times 50. These two parameters yield between 400 and 500 superpixels per image. Different size of superpixels can be tried alternatively.

4.2 Density Count

The goal of this paper is to get a crowd count across frames from videos. In order to get an estimate density count for each superpixel we use our previous method [1]. Our previous method only works in static images. It works by using several sources of information to estimate a count in patches across a frame. A head detection score, Fourier analysis, and interest point counts are used to compute counts for patches. A Markov Random Field is then used to smooth counts from nearby patches and get a total count across frames.

Our previous method outputs a density map which gives a density value to each pixel in the frame. We use the density map to calculate a count per superpixel by adding all the density values of each pixel pertaining to each superpixel. The superpixels counts are calculated across those frames where a count is desired.

4.3 Dense Trajectories

Dense trajectories give trajectories for dense features from a video. They capture the flow of density across frames. We used the code provided by [3] to compute dense trajectories. Since crowds usually show high density these trajectories will capture the direction of motion of crowds. Different number of trajectories can be used for videos. For testing purposes, we decided to use 35,000 trajectories. The number of trajectories was picked arbitrarily but different numbers can be tried as well. Having very few tracks may not capture the motion throughout all points, and having too much trajectories is unnecessary and can become computational expensive.

4.4 Conditional Random Field

Counts calculated by the density map [1] across frames may vary between nearby frames. We use a Conditional Random Field (CRF) to smooth counts across frames. Counts are constrained in both the spatial and time dimensions. Nearby superpixels should have a similar count in absolute frames, while frames connected by dense trajectories across the time dimension should also have a similar count.

The CRF is constructed using the count per superpixels as well as the dense trajectories. Each superpixel across frames will be a unique node in the graph. The value for each node c_{ij} is the density count for that superpixel in which i is the frame number and j is the superpixel label for that i^{th} frame. Nodes are connected with all the neighboring nodes if they fall within a radius r.

The value of *r* can be changed. For experimental purposes we use a value of *r* equal to 2. The binary cost for connected nodes is the absolute difference between the nodes multiplied by an arbitrary *lambda* to serve as a smoothing function.

The dense trajectories are used to connect nodes across the time dimensions. We assume that crowds move in groups and after a certain amount of time that same group will be in a different position. The binary cost for connected nodes across the time dimensions is also the absolute difference between the node values multiplied by the arbitrary *lambda*.

In Figure 3 there is an example and simplified structure of the CRF. All neighboring superpixels are connected by a binary cost. For example, the cost between the first and second node in frame one is $\lambda|c_{11} - c_{12}|$. Let's assume that the crowd sections in n_{11} move to n_{25} and then to n_{39} across the time dimension, *n* being the superpixel label. These three nodes will then be connected in the CRF.

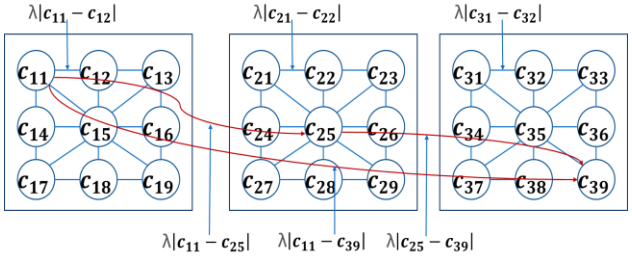


Figure 3: The structure of the CRF. For simplicity purposes not all the connections are shown.

We use [4] to construct the CRF and assign a new count to each node (superpixel). To represent the graph, the method requires a class which specifies the initial label for each class. Since the density values are very small for each superpixel we normalize the values between 0 and 100, giving us a total of 101 possible classes for each superpixel. Another parameter for the function is the unary/potential cost for each of the possible classes at each superpixel. We assign the potential *p* for each superpixel *s* at each of the possible labels *l* by using a normal distribution, with *c* representing the count for the *s* superpixel:

$$p(l, c_s) = 1 - \frac{e^{-0.50*|((l-1)-c_S)^2|}}{\sqrt{2\pi}} \quad (1)$$

Equation 1 will add more probability to the labels nearby the count calculated by the density. The 3-D CRF is represented as 2-D matrix with the binary costs for those nodes connected, and a 0 where there is no connection. Finally, the last parameter is the fixed label cost for the labels of each adjacent node in the graph, in which we just assign the arbitrary *lambda* value. The CRF minimizes and finds the optimal solution that will reduce the cost between superpixels in both the spatial and time dimension by assigning a new count label to each superpixel.

To produce the count on each evaluated frame the labels are normalized back to the regular values. The labels are then added for each frame to estimate the count across frames in the crowd videos.

5. Results

Our proposed method was tested on the dataset described in Section 2. We compared the proposed method to our previous image static crowd counting method [1] to analyze the effect of the CRF for counting in videos. Method [1] was trained with the UCF Crowd dataset of 50 crowd images. For our proposed method to calculate a count per superpixel, we also did the training with the same 50 crowd images. Our proposed method was tested with 400-500 superpixels per image. For simplicity purposes 35,000 dense trajectories are calculated and all of them are kept to construct the CRF. Table 1 shows the results with different *lambda* values.

Method	**λ**	**AD**	**NAD**
CVPR'13	n/a	236.1	30.2
Proposed	1.00	215.6	26.9
	0.65	**214.2**	26.3
	0.60	215.8	**26.2**

Table 1: Results for both our previous method [1] and our new proposed method. Tested on the dataset of videos.

The proposed method performs better in all instances of *lambda* compared to using our static image crowd counting method [1] alone. For the absolute difference we use equation (2).

$$AD = mean(|GTC - PC|) \qquad (2)$$

GTC being the ground truth count from the annotated frames, and PC the predicted count. For the normalize absolute difference metric we use equation (3).

$$NAD = mean\left(\frac{|GTC-PC|}{GTC}\right) * 100 \qquad (3)$$

The proposed method out performs our previous method [1] by 22 people when using a *lambda* equal to 0.65. There is also a 4% improvement as shown in the NAD column with a *lambda* equal to 0.60. In Figure 4 some qualitative results are shown.

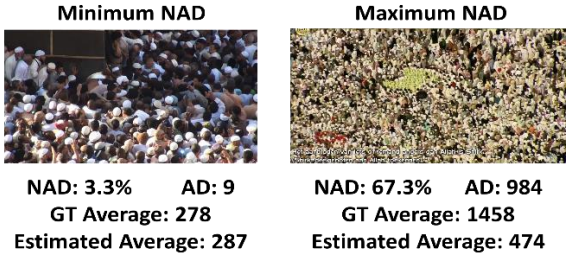


Figure 4: Qualitative results for the proposed method.

6. Conclusion

We have presented a method for counting in videos which improves the results from our previous method that is used for counting in images. The static image crowd counting method is extended by using the extra information of motion to improve counts. The proposed method uses superpixels and dense trajectories to construct a 3-D conditional random field that connects neighboring superpixels so that they have similar counts and we also connect superpixels through dense trajectories to capture flow consistency across the time dimension.

Our proposed method has shown a 4% improvement compared to using [1] alone. Counts are constrained and smooth across frames to have a similar count, therefore, improving the results. The results for the proposed method can improve as the method for counting in static images improves since our proposed method requires a count per superpixel. We have shown that constructing a CRF when using a static image crowd counting method to smooth counts across frames in high dense crowd videos improves the results.

For future work we plan on using the best crowd segmentation techniques to improve counts. By segmenting crowd regions, we can discard counts from non-crowd regions, improving the results in crowd counting in both images and videos.

7. References

[1] Idrees, Haroon, et al. "Multi-source multi-scale counting in extremely dense crowd images." Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. IEEE, 2013.

[2] Vedaldi, Andrea, and Brian Fulkerson. "VLFeat: An open and portable library of computer vision algorithms." Proceedings of the International conference on Multimedia. ACM, 2010.

[3] M. Raptis, I. Kokkinos, and S. Soatto. "Discovering discriminative action parts from mid-level video representations." In CVPR, 2012.

[4] Zabih, Ramin D., Olga Veksler, and Yuri Boykov. "System and method for fast approximate energy minimization via graph cuts." U.S. Patent No. 6,744,923. 1 Jun. 2004.

[5] V. Lempitsky and A. Zisserman. "Learning to count objects in images." In NIPS, 2010.

[6] M. Rodriguez, J. Sivic, I. Laptev, and J. Y. Audibert. "Density-aware person detection and tracking in crowds." In ICCV, 2011.