Background Subtraction using Local SVD Binary Pattern

Franci Suni Lopez

Universidad Católica San Pablo-Arequipa Peru

franci.suni@ucsp.edu.pe

Joel Oswaldo Gallegos Guillen

Universidad Católica San Pablo-Arequipa Peru

joel.gallegos@ucsp.edu.pe

Abstract—**Background subtraction is a technique widely used to detect moving objects from static cameras. In the field of video surveillance, it is essential since it allows rapid discrimination when an event has arisen and studies of mobility of objects on the scene. In the last few years, many methods with different strengths and requirements have been proposed. For instance, background subtraction using local SVD binary pattern was a proposed method combining different techniques.**

In this paper we study, replicate and implement the paper before mentioned, and we compare our results with the paper dataset (CDnet 2012). Our results indicate that LSBP does not work well in environments with shadow and in two datasets (copyMachine and sofa) our implementation are better than original paper.

Keywords-**camera calibration,**

I. INTRODUCTION

Background subtraction is a fundamental stage of a large number of applications in which it is necessary to detect movement, identify and / or follow objects in sequences of dynamic images. Among these applications, video surveillance can be named, such as the detection and capture of movement flow, pose estimation, man-computer interaction or video coding based on content, among others. The process of subtraction is also called foreground extraction or foreground extraction, and consists of a series of methods that allow distinguishing between background or static zones (background), and dynamic zones that correspond to the foreground.

In order to extract the background, a representation of the scene is maintained through the digital modeling of the background. This representation must contain the necessary information for each moment to determine, for each new entry image, which areas of the same correspond effectively with the background. Since the background is not invariant in time, it is necessary to perform a dynamic modeling of the background to adapt it to the variations that may occur in it. The first time, any significant change in a region of the image with respect to the background model will be considered foreground. In general, each element of the model is updated for each certain time interval, as long as it has lost significance.

II. LOCAL SVD BINARY PATTERN

Local Binary Pattern (LBP) is a powerful and fast local image descriptor for analyzing textures. However, LBP is not robust to local image noises. Therefore, the authors add Singular value decomposition (SVD) because is utilized in noise filtering. They presents that SVD coefficients (i.e., singular values) are likely to reveal the illumination-invariant characteristics. Therefore, they try to apply SVD to local regions and they define block B (centered by(x, y)) of MxM pixels:

$$B(x,y) = U \sum V^T$$

where U and V are orthogonal matrixes; $\sum = $ diag $(\lambda_1, \lambda_2, ..., \lambda_n)$ is a nonnegative diagonal matrix with decreasing entries along the diagonal, it is called the singular values of B(x, y). They use blocks of $3x3$ and at each pixel on position B(x,y), they define this scalar for a given frame as:

$$g(x,y) = \sum_{j=2}^{M} \hat{\lambda}_j, \text{ and } \hat{\lambda}_j = \lambda_j/\lambda_1$$

where $\hat{\lambda}_j$ indicates the jth singular value.

The principle of LSBP is to compare a central point value with neighbor values and check whether they are similar or not. And local structural invariant is applied for central point value and neighbor values. Texture at point (x_c, y_c) is modeled using a local neighborhood of radius R, which is sampled at P points [?] [?]. The LSBP binary string at a given location (x_c, y_c) can be derived from the following formula:

$$LSBP(x_c, y_c) = \sum_{p=0}^{p-1} s(i_p, i_c)2^p$$

Algorithm 1 Background Subtraction for FG/BG segmentation using LSBP feature.

Initialization:

1: **for** $each\ pixel\ of\ the\ first\ N\ frames$ **do**
2:    Extract the LSBP descriptor for each pixels using Equation (12)
3:    Push color intensities into $BInt_{index}(x,y)$ and LSBP features into $BLSBP_{index}(x,y)$ as the background model
4:    Compute $\overline{d}_{min}(x,y)$ for each pixel.
5: **end for**

Mainloop:

6: **for** $each\ pixel\ of\ newly\ appearing\ frame$ **do**
7:    Extract $Int(x,y)$ and $LSBP(x,y)$
8: **end for**
9: $matches \leftarrow 0$
10: $index \leftarrow 0$
11: **for** each pixel in current frame **do**
12:    **while** $((index \leq N)\ \&\&\ (matches < \sharp min))$ **do**
13:      computer $L1dist(Int(x,y), BInt_{index}(x,y))$ and $H(LSBP(x,y), BLSBP_{index}(x,y))$
14:      **if** $((L1dist(x,y)\ <\ R(x,y))\&\&(H(x,y)\ \leq\ H_{LSBP}))$ **then**
15:        $matches\ +=\ matches$
16:      **end if**
17:      $index\ +=\ index$
18:    **end while**
19:    **if** $(matches < \sharp min)$ **then**
20:      $Foreground$
21:    **else**
22:      $Background$
23:    **end if**
24: **end for**

---

where i_c is the central point value, i_p represents the N-neighborhood point value. τ is the similarity threshold which is set to 0.05 in the original paper [**?**]. $S(.)$ is a sign function defined as follows:

$$s(i_p, i_c) = \begin{cases} 0 & \text{if } |i_p - i_c| \leq \tau \\ 1 & \text{otherwise} \end{cases}$$

Finally, to classify a pixel at coordinate (x,y), the current frame should be matched against their samples. The pixel value $Int(x,y)$ and $LSBP(x,y)$ value are both need to be matched correctly . We call this combined verification.

$$(H(LSBP(x,y), BLSBP_{index}(x,y)) \leq H_{LSBP})$$

$$\&\&(L1dist(Int(x,y), BInt_{index}(x,y)) < R(x,y))$$

$\#min$ is the minimum count of matches needed for classification. They fixed $\#min = 2$ because it is a reasonable balance between noise resistance and computational complexity. For LSBP comparison, they use Hamming distance (XOR) and they select simpler L1

distance for the color intensity comparison. Algorithm 1 presents the background subtraction for FG/BG using LSBP descriptor [1].

III. Paper limitations

Figure III presents a problem with the comparison of color using the L1 distance, where the color of subject pants are confused with the sofa color, this is due to the colors are similar but not equal and using L1 distance it takes as the same color.
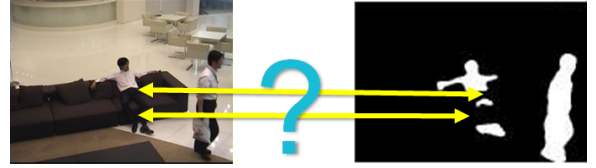

Fig. 1. Wrong segmentation due to color comparison (L1 distance).

Then whether we change the value of the color threshold we can obtain the segmentation problems shown on Figure III, where whether we use a low threshold the algorithm could differentiate between color (brown of black) but we obtain much noisy (left image of Figure III), in opposite case whether we use a high threshold we algorithm could not differentiate between similar colors (right image of Figure III). Therefore with a balanced threshold we could obtain a better segmentation of the background (central image of Figure III).
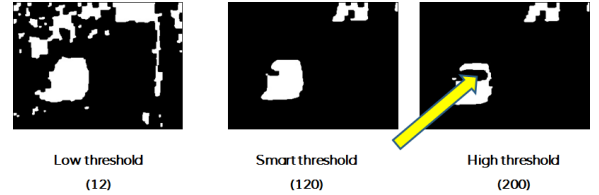

| Low threshold (12) | Smart threshold (120) | High threshold (200) |

Fig. 2. Results with different color threshold.

IV. Our Implementation

According to the limitations before mentioned, for the color difference we propose to change the color space to Lab, because it differentiates better between color and instead to use L1 distance we use CIEDE2000 distance [1]. Then, over the difference image we apply two morphological operations (erode and dilate) to reduce the noise of the image. Finally, to solve the holds caused by the color difference we use a fill contours function. Figure IV summarizes our changes based on the original proposal.

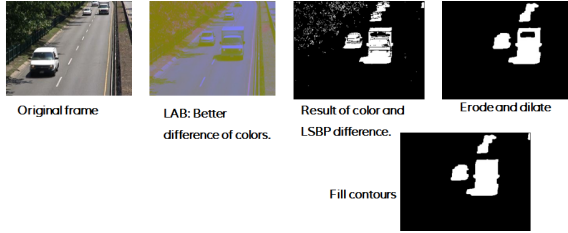[1] $https : //en.wikipedia.org/wiki/Color\_difference\#CIEDE2000$

Fig. 3. Our changes of the original paper.

V. LSBP SOURCE CODE DESCRIPTION

The source code was implemented on ***python***, therefore we need the python compiler installed on our computer. Initially, we installed the the prerequisites on Ubuntu 16.04, but we can not run the code because there was a problem of compatibility between the *numpy* and *maxflow* libraries. Then we install Anaconda2 version 4.4.0 on windows 7 and also install *maxflow* library manually. Finally we run the source code with the following script:

```
python local_svd_2016
−g  highway\groundtruth
−f  highway\input
−k  1700
```

Where *-g* is the Directory with ground-truth frames, *-f* is the directory with input frames and *-k* calculates answer for K-th frame and output.

VI. RESULTS

We use the CDnet 2012 dataset[2] to test the source code and our implementation. From them we take the five datasets that are showed on the original paper (*copyMachine*, *highway*, *overpass*, *PETS2006* and *sofa*). Figure 5 presents the original frames, the GroundTruth, the paper results, source code results and our implementation results. Where in comparison with the source code time

our implementation is better because in average our implementation takes 164.8 ms to compute each frame and source code takes in average 4.81 seg by frame. Additionally our results are more similar to paper results and in two datasets (copyMachine and Sofa) our result are better than paper. Finally, we do a comparison of results with our own data set with the source code, Figure **??** presents the selected frames and the results of both proposals. As is shown on Figure **??** our results are better than source code in time and in background segmentation.

[2]http://jacarini.dinf.usherbrooke.ca/dataset2012/

VII. CONCLUSIONS

In this paper we present a comparison between three works of Background Subtraction Using Local SVD Binary Pattern (i.e. original paper, source code and our implementation). Our results present that in two datasets (copyMachine and Sofa), our implementation are better than paper results. Also, in comparison with source code our implementation are better in time and background segmentation. Finally, we found that original color comparison (with L1 distance) generates problems with similar colors and the LSBP are not a good descriptor to solve shadow cases.

REFERENCES

[1] N. Goyette, P. M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection.net: A new change detection benchmark dataset. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, June 2012.
[2] Thierry Bouwmans. Traditional and recent approaches in background modeling for foreground detection: An overview. *Computer Science Review*, 11-12:31 – 66, 2014.
[3] G. A. Bilodeau, J. P. Jodoin, and N. Saunier. Change detection in feature space using local binary similarity patterns. In *2013 International Conference on Computer and Robot Vision*, pages 106–112, May 2013.
[4] L. Guo, D. Xu, and Z. Qiang. Background subtraction using local svd binary pattern. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1159–1167, June 2016.
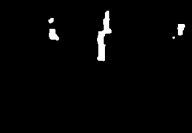
	Highway	**Copy Machine**	**PETS2006**	**Overpass**	**Sofa**
Frame					
GroundTruth					
Paper					
Source code					
Time by frame	4.19 sec	5.12 sec	5.28 sec	4.97 sec	4.49 sec
Our implementation					
Time by frame	169 ms	157 ms	175 ms	159 ms	164 ms

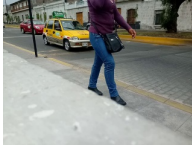Fig. 4. Comparison between original frames, GroundTruth, results of paper and our results.

Street-people   **Street-cars**

Frame

Source code
Time by frame   5.19 sec   5.19 sec   6.23 sec   6.23 sec

Our implementation
Time by frame   165 ms   165 ms   157 ms   157 ms

Fig. 5.   Comparison between original frames, source code and our implementation of two recorded videos.

5