Maestría en Ciencias de la Computación
Background Subtraction using Local SVD Binary Pattern

Diego Alonso Javier Quispe , David Choqueluque Roman
Universidad Católica San Pablo
{ diego.javier, david.choqueluque}@ucsp.edu.pe
Arequipa,Perú

Abstract— **Background substaction is a basic problem for change detection. The main problem of detect changes are the illuminations changes. For solve this problem the propose is use LBP pattern and use SVD desomposition because the singular values are invariant to the illumination. We developed the algorithm in c++ one implementation using threads and other implementation using cuda to optimize the processing time. We use the CDnet 2012 dataset to prove the implementation.**

In the next sections we show the general framework of the method, the seudocodigo of Local SVD Binary Pattern and the results of apply the method in differents scenarios.

Key words: Local SVD Binary Pattern, Cuda, Threads

I. INTRODUCTION

The background subtraction has many applications in the real live. The background scene is discarded and you can analyze only the objects in movement.

II. MODELING BACKGROUND USING LOCAL SVD BINARY PATTERN

In the **fig. 1** you can see the overview of the framework described in [1].

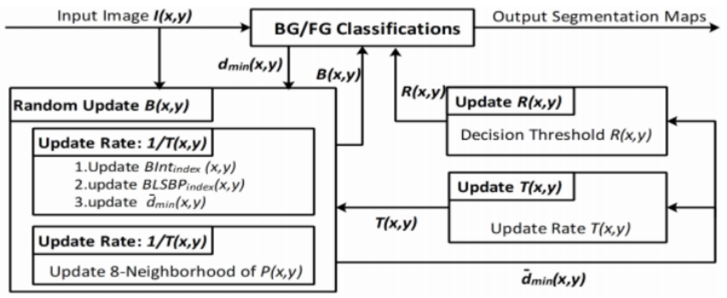
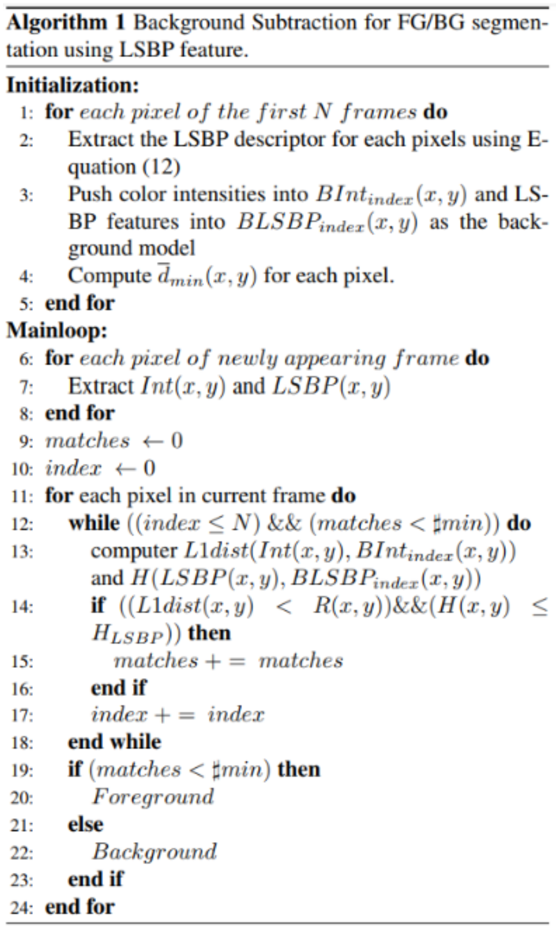


Fig. 1. General Pipeline.

In **fig 2.** you could see the pseudocodigo of Local SVD Binary Pattern. The improvement implemented is where the LSBP of a frame is extracted, in that step we use threads in one implementation and cuda in the second implementation to reduce the processing time.



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
```

Fig. 2. General Pipeline.

III. Experimental results

For reduce the processing time we develop two improvement.
Firtsly we use threads for the LSBP descomposition of a frame, we use a total of 4 cores. For obtain the singular values of the SVD descompotition que use the library **armadillo**.
Secondly we use cuda for the LSBP descomposition of a frame, for obtain the singular values of the SVD descompotition we use a SVD implementation in CUDA.
The comparation of processing time of the python implementation with the the two implementation described is shown in **Table I**.

	Frame	Python	Threads	Cuda
highway	240x320	5.508s	0.864s	0.484s
office	240x360	5.858s	0.980s	0.563s
peopleInShade	240x380	6.337s	1.096s	0.587s
streetLight	240x320	5.131s	0.847s	0.493s
own video	270x480	-	1.652s	0.836s

TABLE I
Comparation of processing time per frame.

There is a moment where the camera slightly moves and generates a noise where there are sections without movement, it is necessary to indicate that this noise disappears quickly with the iterations, you could apreciated it in **Fig. 3**.

In the **Fig. 4** you could apreciated a scene with noise because the video of this scene was taken with a cell phone and it was not fixed so it generates a lot of noise when detecting the movement in the scene as a change.

Fig. 3. Noise in CDnet 2012.

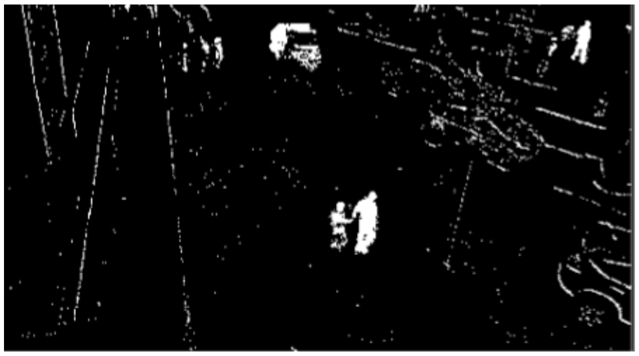


Fig. 4. Noise in our video.

In **Table II** it can be apreciated the comparation of the results of the implementation in python and c++ in differentes scenarios of the CDnet 2012 dataset. We use the scenarios titled **highway**, **office** and **street highligth**. Visually our results have less noise than the implementation in python.

In **Table III** a video taken at the catholic university san pablo was used, where people walking and cars moving can be seen.

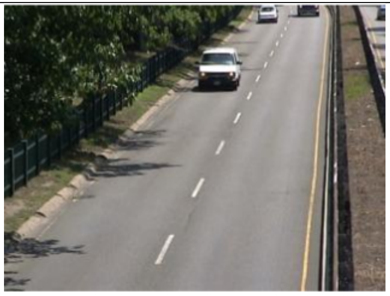
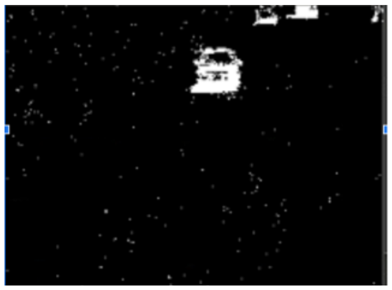
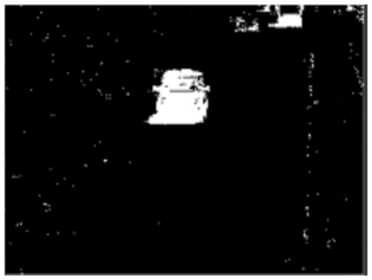
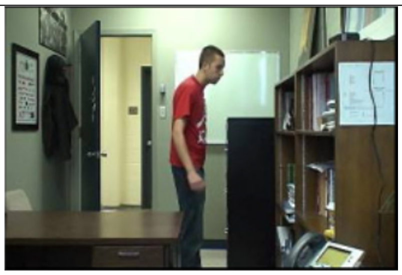
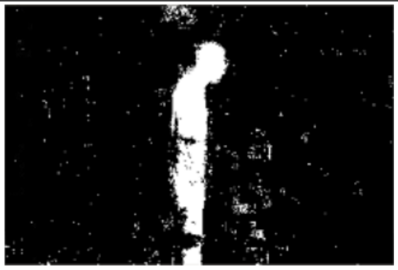
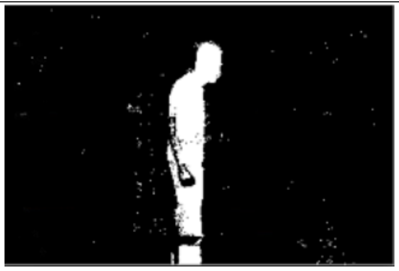
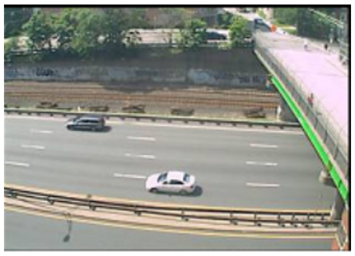
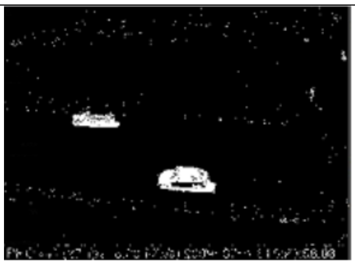
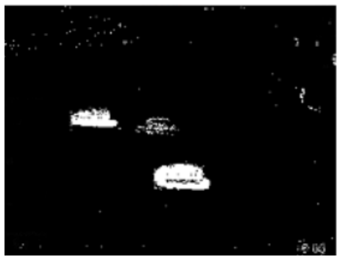
	Input	Python	Our Implementation
highway			
office			
street			

TABLE II
CDNET 2012.

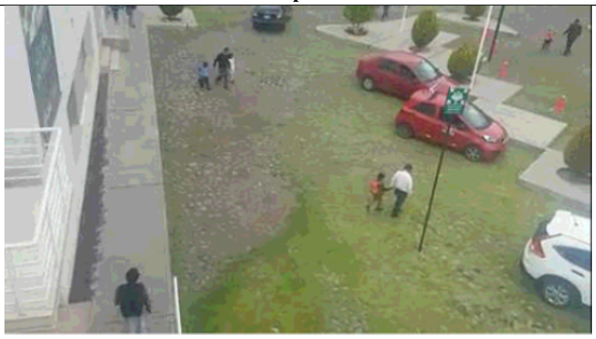
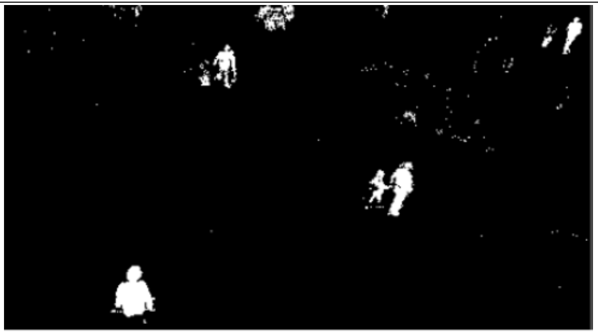
	Input	Our Implementation
highway		

TABLE III
UNIVERSITY SAN PABLO VIDEO.

IV. Conclusions and Future Work

We implemented an improvement to obtain the LSBP of a frame using threads and cuda. Compare with the implementation in python, our implementation in c++ reduce the time per frame in a proportion of 6:1 using threads and 11:1 using cuda. We reduce noise by identifying and correcting errors in the python code. When the camera have little movements this generates noise in the result. We obtain better results comparing with the implementation in python but no better than the results of the autor.

References

[1] Lili Guo, Dan Xu, Zhenping Qiang,"Background Subtraction using Local SVD Binary Pattern", 2016.

[2] M. Hofmann, P. Tiefenbacher, and G. Rigoll, "Background segmentation with feedback: The pixel-based adaptive segmenter", 2012.