

Background extraction with a coarse-to-fine approach

Eric Thea, ethea@stanford.edu

Abstract – Extracting the background from a video sequence is an open problem with very practical applications ranging from camera surveillance systems to human-computer interactions. It is a difficult problem because a practical algorithm needs to account for various phenomena such as illumination changes, background object displacements and non-static backgrounds amongst other things. Moreover, because the background extraction has to be done in real time, the method has to be computationally efficient yet robust enough. In this work, we propose a coarse to fine approach to address this problem. Specifically, we use a Gaussian mixture model to describe the color at each pixel location. But the classification process is first done at the block level according to a proportion criterion. In the regions that have not been classified as background after this first step, we perform the classification at the pixel level. This method can allow the algorithm to be faster than the pixel-only approach for the same visual performance.

1. Introduction

Background extraction is the process of distinguishing novel (foreground) from non-novel (background) elements in a scene from a video sequence.

Because this process can be used in many different systems and under very different conditions, the algorithm constraints can be very different from an application to another. An outdoor environment will for example have very different characteristics compared to an indoor environment. Similarly, an application could require very precise detail capturing whereas movement detection would be sufficient to another application.

But we can nonetheless specify two characteristics that we would like to find in any algorithm: real time processing and real environment performance. This implies that some sub optimal methods will sometimes be used to perform the decision classification. This also means that various effects will degrade the algorithm performance unless they

have been accounted for. These include moving object shadows that appear on background elements, illumination changes due to natural or artificial lighting, background object displacements, non-static background such as computer screen flickers and so on.

In this work, we use the models described in [1] [2] but apply a block method in addition to a selective pixel approach. This allows the algorithm to be run faster while maintaining the same visual performance as the one achieved with a pixel procedure.

2. Pixel model

Similarly to [1] [2], the color at each pixel location is described by a mixture of K Gaussians. The Gaussian representations account for the presence of noise in the measurements. Using K of them is a good way to efficiently describe changes in a non-static scene while not discarding previously learned information. It also allows us to model several existing backgrounds simultaneously: consider for example tree leaves in the wind that sometimes occult the sky and sometimes not. Each Gaussian is obviously characterized by its mean and its variance, but we also need to indicate its relative weight in the background description. Because the algorithm needs to adapt to modifications in the scene, we will use the observations at each time to update these three parameters. Changes will be incorporated to the model according to a learning rate.

The algorithm itself works the following way: for any given frame, we start with determining for each pixel the Gaussian that matches its value best. Real time constraints impose us to use a sub optimal on-line K-means approximation to perform that selection. The idea is to choose the Gaussian for which the difference between the pixel value and the Gaussian mean is smaller than a fixed constant times the Gaussian standard deviation. If no match is found, we classify the pixel as foreground and we create a new Gaussian to replace the least probable one. This new Gaussian will have a mean equal to the pixel value, a high variance and a low initial weight.

In the case a match was found, we need to determine whether this Gaussian corresponds to the background description or not. We first sort the Gaussians according

to the ratio between their weight and their standard deviation. A Gaussian representing a background element will have a high weight and a low variance and will therefore appear first. A Gaussian representing a foreground element will have a low weight and a large variance (due to movement) and will therefore appear towards the end of the ordered sequence. We then create a set that contains the first Gaussians for which the sum of their weights is higher than a threshold T . This set represents the background model. If the matched Gaussian found earlier belongs to this set, the pixel is classified as background. If not, it is classified as foreground.

We finally update the parameter values according to the formula in [2] and go to the next frame.

3. Block model

We now incorporate the pixel model into a broader framework that will support block manipulations. The motivation behind a block approach is that there are some possibly large regions in the frame that don't change much over time so they can easily be classified as background. For such regions, the update process could also be less frequent, saving on some computation capacities. Another reason for dealing with blocks is that all elements in the scene really have spatial extension, they have a shape (although this could be varying) and move according to identifiable patterns. Going to a block level is a way to "zoom out" and this could allow us to capture these abstractions. In this work, we won't implement any shape model or motion estimation technique, but these are possible extensions to our project.

We adopt a coarse-to-fine approach by first dividing the frame into blocks. Using the same Gaussian mixture model as above to describe each pixel, we calculate in each block the proportion of pixels that are classified as background. If this fraction is larger than a threshold value, the whole block is classified as background and is not processed further. If this fraction is lower than the threshold, we then go to the pixel level within this block and run the pixel algorithm described in the previous section.

This algorithm averages out the noise in the block regions and therefore gives more robust results. It can also help us save some computation power as some pixels are simply discarded after the block level classification and not processed further. But the disadvantage of such technique is the apparition of some blocking artifacts in the extracted frame.

4. Experimental results

Let us first describe the parameters used in the algorithm and the implementation choices that were made.

We decided to work in the YUV space rather than in the RGB domain because it experimentally appeared that this reduces shadow misclassification. We described each pixel with 3 Gaussians (some of them could be unused) and the initial variance was set high enough to prevent noisy pixels in the background from being classified as foreground. The threshold value T was set equal to 0.4 so that one or two Gaussians would usually account for the background.

We used a learning rate equal to 0.006. This is a relatively low value that prevents the algorithm from learning too fast and therefore limits shadow effects.

We chose 16 by 16 blocks and also implemented a small connectivity procedure to reduce the number of holes in the foreground elements.

The results are mostly satisfying. The visual quality obtained with the block approach is very similar to the quality resulting from a pixel method. Note that we use a visual criterion to compare the two methods as it is impossible to compute the MSE or PSNR for such an application. This algorithm was also faster than the pixel approach, for the reasons described above.

But there are also some undesirable effects as we can notice some blocking artifacts in the extracted frame. Moreover, some specific phenomena such as non-static backgrounds are not well addressed with this current algorithm.

5. Conclusion

In this work, we presented a coarse-to-fine approach to background extraction. We used a Gaussian mixture model similar to [1] [2] but applied it to a two step procedure. This allowed us to perform the classification at the block level and then at the pixel level. This resulted on some computation savings while keeping the same visual quality.

A possible extension to this work would completely dissociate the pixel level and the block level. Different Gaussian models could for example be used to represent the block and the pixels within the block.

Other possible extensions include the implementation of a motion estimation algorithm or the use of shape models to represent foreground elements.

Background extraction is still an open problem that has to be thought with the application in mind. Necessary trade offs are to be made to come up with a satisfying solution.

Acknowledgments:

We would like to thank Michael Harville, Susie Wee and John Apostolopoulos for their help during the course of this project.

References:

- [1] Harville, M.; Gordon, G.; Woodfill, J., "Foreground segmentation using adaptive mixture models in color and depth," *Detection and Recognition of Events in Video, 2001. Proceedings. IEEE Workshop on* , 8 July 2001.
- [2] Stauffer, C.; Grimson, W.E.L., "Adaptive background mixture models for real-time tracking," *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on* . ,Volume: 2 , 23-25 June 1999.

Appendix

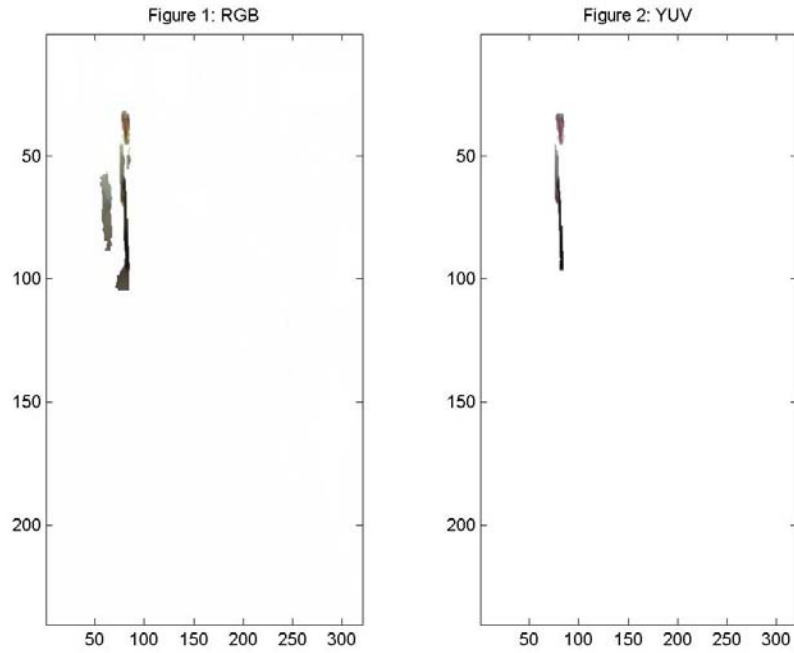


Fig 1-2: Shadow misclassifications are reduced when working in the YUV domain.

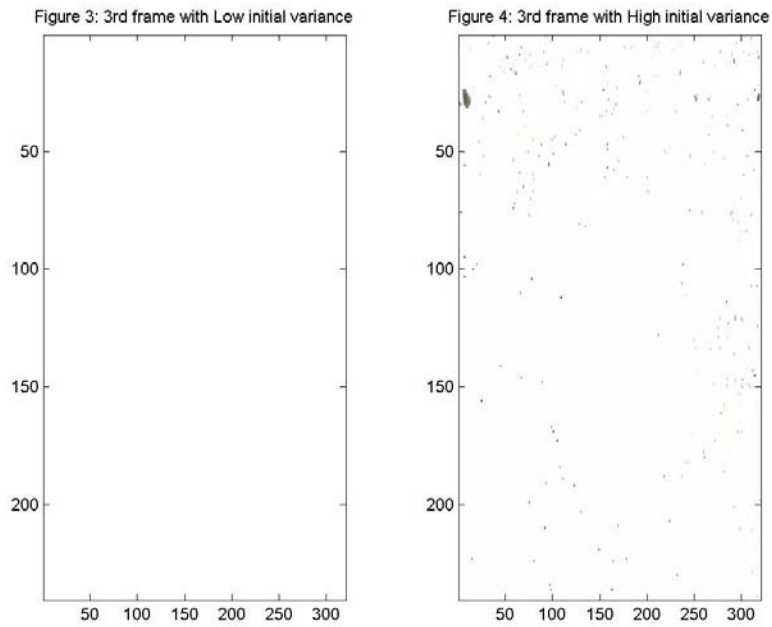


Fig 3-4: The background appears noisy when the initial variance is set too low.

Figure 5: RGB, high variance, low learning rate

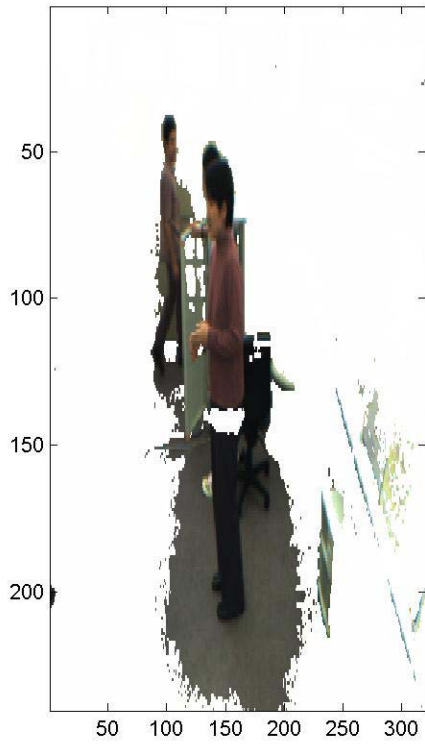


Figure 6: RGB, low variance, high learning rate

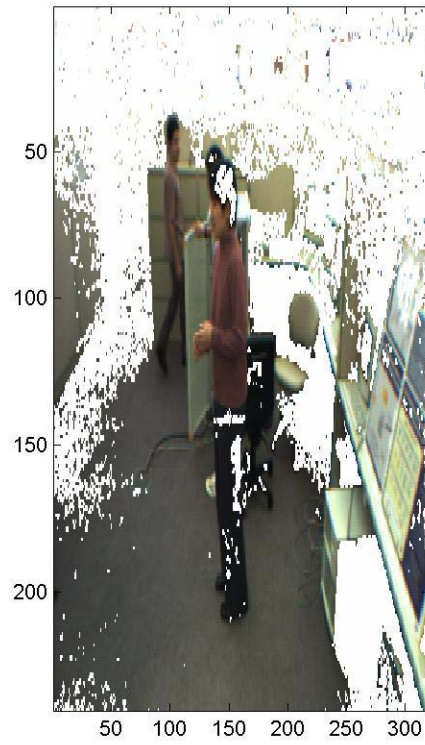


Fig 5-6: The noisy background is due to the high initial variance, the high learning rate creates the shadow misclassifications (more background is visible).

Figure 7: Block based approach

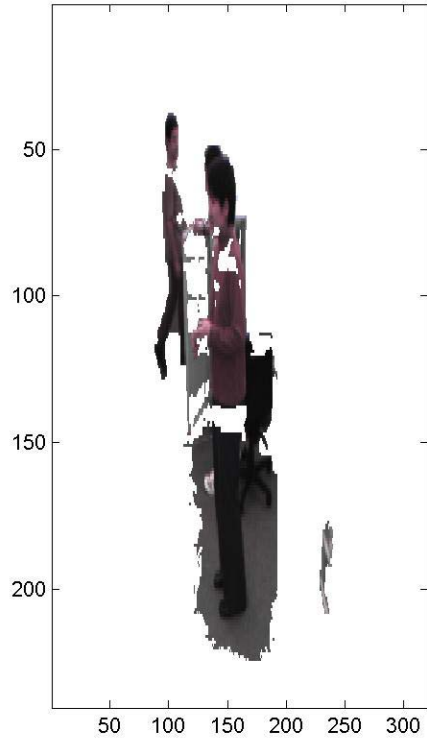


Figure 8: Pixel based approach

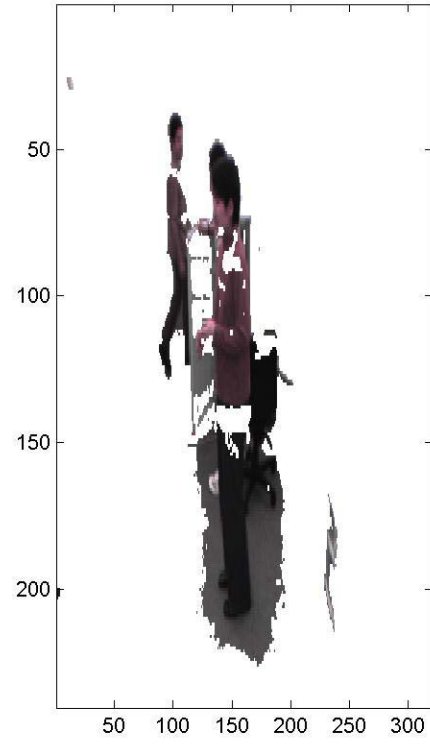


Fig 7-8: The block approach and the pixel approach give similar visual results