Dynamic Foreground/Background extraction based on segmented image

Huimin Yang
Electrical and Computer Engineering
Carnegie Mellon University
huiminy@andrew.cmu.edu

Tianyi Chen
Information Networking Institute
Carnegie Mellon University
tianyic@andrew.cmu.edu

Yulian Xu
Electrical and Computer Engineering
Carnegie Mellon University
yulianx@andrew.cmu.edu

Ran Ye
Electrical and Computer Engineering
Carnegie Mellon University
rany@andrew.cmu.edu

Abstract

This paper addresses the problem of extracting dynamic foreground regions from a relatively complex environment within a collection of images or a video sequence. By using image segmentation code, we can first convert our traditional pixel-wise image collection into a collection of image with multiple monochrome image segments. Our approach in this study consists of four steps. First of all, we uniformly extract patches from the first frame of segmented image collection. And then, we manual tag the foreground and background within the first segmented image. After this step all the patches in the first frame will form two bags of patches. For one bag, the patches in it can model the features of the foreground. Meanwhile, the patches in the other bag can describe the features of the background. In this case, we call them foreground patches and background patches respectively. Third, for an incoming frame, we perform the segmentation and then extract the patches. For both the patches from the new frame and the previous known patches, in order to reduce the dimension, we perform LDA. Then use KNN and KDE to find the nearest patch bag for the incoming patches. At this point, we can differentiate the foreground and background for the new image frame. Finally, we perform a bidirectional consistency check between the patches we already get and the patches from incoming image frame make the model adapt to new incoming image frames. In this report, we practice a novel, clear and easy way to extract dynamic foreground from the complex background.

1 Introduction

In this project, we propose to extract foreground/background based on segments. Traditionally, when the camera is still, the typical methods of separating foreground from background are based on the fact that foreground objects are moving and background remains unchanged. In such scenario, foreground/background separation can be achieved by detecting the change across frames of the video. Pixels that changes their position across the frame could be classify as foreground, while pixels remains the same across frames can be classified as background.

However, when the camera is moving, the traditional approach of detecting changes across frames could not give us satisfying result because both foreground and background objects are moving. Therefore, we propose to extract separate foreground/background image by the feature of a segment. In a segmented image, if the feature of a segment is closer to the feature of previous foreground segments, we classify it as a foreground segment. Otherwise, we classify it as a background segment.

There are multiple ways to represent the feature of a segment. In this project, we use 2 approaches to represent the feature of a segment – raw RGB vector, LDA feature.

Since a segment is a continuous area in the image within which color and texture are similar. Therefore, we propose to extract patches within a segment, and use the patches to represent the segment to reduce the computational complexity. The patches from previous frames form a foreground patch bag and a background patch bag to be compared against for new patches. And the bags adaptively update and evolve as new frames get processed.

The rest of the report is organized as below: Section 2 describes the representation of patches and segment and the approach that we use to determine the similarity of patches and segments. Section 3 presents the algorithm and theoretical analysis in detail. Section 4 presents our experiment result.

2 Patch representation and matching

Because color and texture within a segment is similar, the feature of a patch within a segment can be used to represent the feature of segment. In this project, we use 3 approaches to represent the feature of a patch.

2.1 Patch Representation

2.1.1 Raw RGB vector.

Because we use patches to represent segments and the size of the patch can be controlled within a reasonable limit. Raw RGB vector can be used as a feasible feature of a patch. For a video of 1920x1080 video, the typical size of a patch is 10 – 20 pixels. Therefore the length of a raw RGB vector is 300 – 1200. The speed of computation of using raw RGB vector is not high, but is still acceptable.

2.1.2 LDA feature of raw RGB vector

Linear Discriminant Analysis (LDA) which also named Fisher Linear Discriminant (FLD) is a classic algorithm in pattern recognition. LDA can project high dimension pattern samples to the optimal discriminant vector space. After the projection, new samples have the maximum values of between-class scatter matrix S¬B and minimum values of within-class scatter matrix S¬W that means the samples have the vest separability in the space. Therefore, LDA is an effective feature extraction method.

2.2 Patch Matching

After we represent a patch, we have to decide whether the patch is more similar to the patches in the foreground bag or the patches in the background patch. This process is called patch matching. In this project, we use KNN to find the nearest neighbor of an un-identified patch.

3 Algorithm

In this section, we first introduce the basic steps of the algorithm. And then we provide a detail analysis on the LDA feature representation of patches. We also give details of patch update process.

3.1 Algorithm

In order to simplify the description of the extraction process, the following terms are defined:

Pf	It denotes the group of patches which are generated from the frame “f”, f = 1, 2… F.

Bag_f	It denotes the bag of foreground patches, which is also called "foreground model".
Bag_b	It denotes the bag of background patches, which is also called "background model".
Sf	It denotes the segments of frame "f", f = 1, 2… F.
Tf	It denotes the tags (foreground/background) for the segments of frame "f", f = 1, 2… F.

The detailed the algorithm (extraction process) is as follows:

1. Segment all the frames of the video, so we get a series of Sf, f = 1, 2… F.

2. Manually tag the foreground and background within the first segmented frame S1, store the tags as T1.

3. Uniformly extract patches P1 from S1. Let's say current frame is the first frame, so f = 1;

After this step, all the patches P1 will form two bags of patches "Bag_f" and "Bag_b" for foreground and background, according to the tags T1 of all the segments S1.

4. The start of extraction process for the incoming new frame f = f + 1.

5. Uniformly extract patches Pf from Sf. Using "KNN" or "KDE" methods to match all the patches in Pf with the two model-patch bags "Bag_f" and "Bag_b". If the patch is very large, we need to use the "PCA" or "LDA" methods to do a dimension reduction before the matching process.

6. For each segment in Sf, tag it to either foreground or background segment based on the voting result of all the patches in this segment. Say each patch in this segment will vote based on the result of step 5.

After this step, all the segments Sf will have their own tags for foreground/background, which means we've successfully extracted the foreground/background from this frame f. Store the tags as Tf.

7. Update the two model-patch bags:

1) Based on Euclidean distance, do a forward consistency check of the new extracted patches Pf against the model patches. Remove those redundant, ambiguous and possible outlier patches from Pf. Then integrate the filtered Pf to the model-patch bags "Bag_f" and "Bag_b".

2) Do a backward consistency check of the two model-patch bags against the new extracted patches Pf. Remove those most out-of-date patches from "Bag_f" and "Bag_b", until the sizes of the two model-patch bags are in the desired range.

Here, what we need to pay attention to is, since we don't want to mistakenly remove the modes of the appearance of the foreground/background image, we need to perform a partition of the patches in the two model-patch bags, before we do the backward check and remove out-of-date patches. Else, we may remove too much patches from some modes, which may lead to poor quality of later classifications. And we can use like "k-means" method or "hierarchical classification" technique to do the modes partition.

8. If this is not the last frame of the video, go to step 4. Else, we are done.

3.2 LDA analysis

3.2.1 Details of Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) which also named Fisher Linear Discriminant (FLD) is a classic algorithm in pattern recognition. LDA can project high dimension pattern samples to the optimal discriminant vector space. After the projection, new samples have the maximum values of between-class scatter matrix S_B and minimum values of within-class scatter matrix S_W that means the samples have the vest separability in the space. Therefore, LDA is an effective feature extraction method.

In our project, we use LDA in order to reduce dimension of pixel sequence of each patch. Consider there are only 4 different classes but each pixel sequence has 1083 dimensions, we need to calculate the 3-dimension eigenvector W standard for each letter.

The mean value of class i is

$$u_i = \frac{1}{n_i}\sum_{x\in i} x \qquad (3)$$

The mean value of training dataset is

$$u = \frac{1}{m}\sum_{i=1}^{m} x_i \qquad (4)$$

Between-class scatter matrix S_B and within-class scatter matrix S_W are

$$S_b = \sum_{i=1}^{c} n_i (u_i - u)(u_i - u)^T \qquad (5)$$

$$S_w = \sum_{i=1}^{c}\sum_{x\in i} (u_i - x_k)(u_i - x_k)^T \qquad (6)$$

The traditional method is using (3), (4), (5), (6) as input into

$$J_{fisher}(\varphi) = \frac{\varphi^T S_b \varphi}{\varphi^T S_w \varphi} = \frac{\sum_{i=1}^{c} n_i \varphi^T (u_i - u)(u_i - u)^T \varphi}{S_w = \sum_{i=1}^{c}\sum_{x\in i} \varphi^T (u_i - x_k)(u_i - x_k)^T \varphi} \qquad (7)$$

In our program, we only need to use the [W, lambda] = eig(S_b, S_w) function in matlab to retrieve the eigenvector W. After multiple the original dataset to W, we get new dataset which is 3 dimensions in each row.

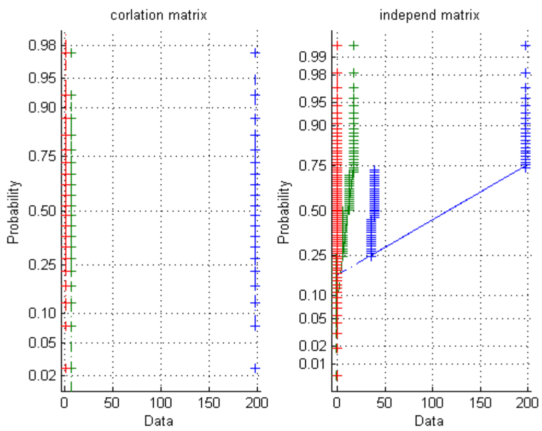

Figure 1 The eigenvector of 3rd segment compare with random patches with same amount when there are four different segment classes

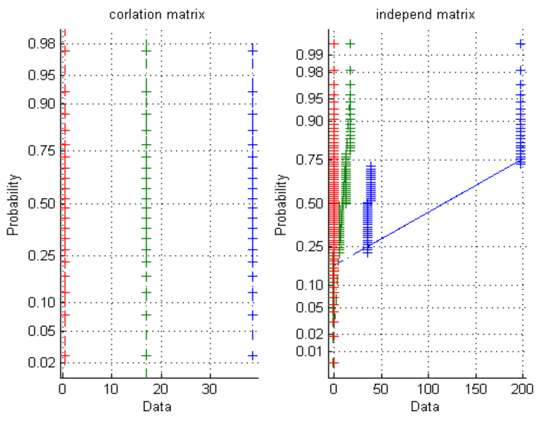

Figure 2 The eigenvector of 4th segment compare with random patches with same amount when there are four different segment classes

We provide two examples to illustrate why our eigenvector should make sense. The left part of each figure is the result of patches in the same class and the right part is the result of random patches. There are three important features to confirm the eigenvector is correct:

1. Each dimension of patches in the same class is parallel with others and is a line themselves means they nearly have same value as their eigenvalues.
2. The left part is more separable than the right part which means each dimension can better represent data features.
3. Different color standard for different dimension. We can see there are a lot of differences between two left figures which means the eigenvector can distinguish different segment well.

3.2.2 KNN Classifier

We implement the KNN as the method of classifier.

We can define the Euclidean distance and the angle between two vectors. For Euclidean distance, we use the equation:

$$\text{distance } (x_1, x_2) = \sqrt{\sum_{i=1}^{n} (x_{1,i} - x_{2,i})^2}$$

In our project, we only need to find the nearest neighbor which means K = 1.

4 Experiment

We use the representation described in Section 2 and implement the algorithm in Section 3. We conducted foreground background extraction on 3 sets of video. In the first set of video, the background is very simple and clean and the foreground object takes up a large portion in the image. In the second set of videos, the background is more complicated but is very different in color and texture with the foreground object. In the third set of videos, background is extremely complicated and the color and texture of the background is similar to part of the foreground image. The result of the experiment on each of the set of video is shown below.

For foreground/background extraction in the case where the camera is moving, there does not exist an automatic way of calculating error rate. In order to calculate the exact error rate, the only way is to manually tag which part of the image belongs to the foreground or background for every frame in the video. Due to the limit time of our project, we are not able to do foreground/background separation for every frame and therefore cannot provide the evaluation in the form of error rate. Instead, we show the some resulting image of foreground/background extraction for each experiment to show the performance of our algorithm.

1. Simple background
   For video of simple background, the result of foreground background extraction is shown below.

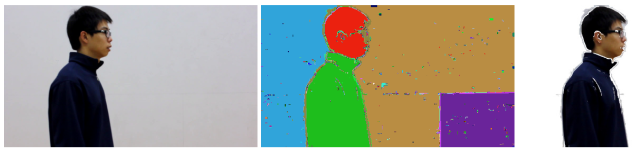

Figure 3 Sample result of simple background video set

number of segments	150 - 250
patch size	20x20 pixels
max patch # per segment	20
minimum segment size	10000 pixels

Table 1 Parameters used for simple background condition

In this set video, the result of using the 3 different approach of patch representation is very similar and all of them produce very good results.

2. More complicated background
   In the second set of videos, backgrounds are more complicated, but the texture and color is different than the foreground.

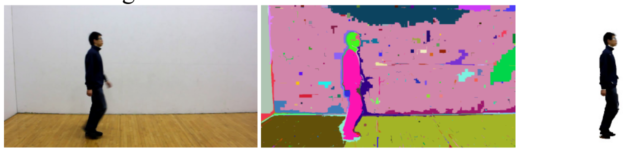

Figure 4 Sample result of comparatively complex background video set

number of segments	250 - 350
patch size	15x15 pixels
max patch # per segment	20
minimum segment size	5000 pixels

Table 2 Parameters used for comparatively complex background condition

In this set of video, the result of using Raw RGB vector representation of patches is the best. For LDA feature representation, some segments are wrongly classified occasionally in some frames.

3. Extremely complicated background
   In the third set of videos, backgrounds are extremely complicated, and the texture and color of the background may be very similar to part of the foreground.

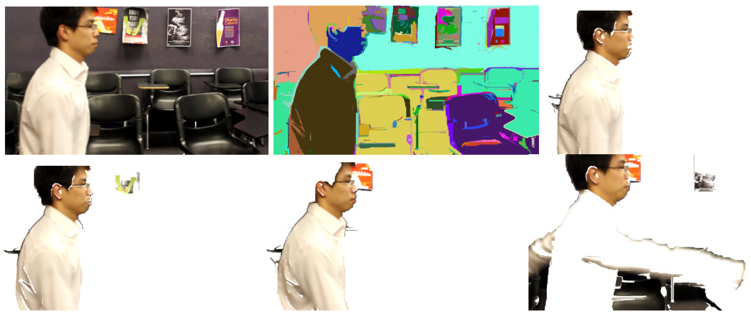

Figure 5 Sample result of extremely complex background video set

number of segments	500 - 1000
patch size	15x15 pixels
max patch # per segment	50
minimum segment size	5000 pixels

Table 3 Parameters used for extremely complex background condition

As shown in the result, because some part of the background is similar to the foreground, several segments are wrongly classified. For example, both the chair and the person's hair are black. As a result, the chairs are sometimes classified as foreground while they are actually background, and the person's hair are sometimes classified as background while they are actually foreground. The error could be reduced if in the video more lights hits on the person's hair and show the texture of the hair. In that case, because the texture of the hair and the texture of the chair are different, the error in foreground/background separation should be reduced.

5 Conclusion

We have just presented our simple but novel algorithm for background/foreground extraction from a series of video frames. While both the foreground and background objects are keeping moving and dynamically changing their appearance during the whole video, according to our experiment results, it is demonstrated that our algorithm can work very well for either simple or complex background. And except the first hand-tag frame, all other frames' extractions are automatically generated based on segmentation and auto appearance learning and matching.

This algorithm employs the "model patch bags" to represent the model background/foreground appearance. To keep the "model patch bags" more adaptive to the changing appearance of the video frames, and also make sure the size of the "model patch bags" is in a reasonable range for calculating, online "model patch bags" updating is realized by adding new qualified patches from new video frames and removing those out-of-date model patches. This makes the algorithm work very reliable and robust.

While the experiment results are satisfying, we can still make progress and improve our performance by considering the spatial relationships among different segments of the foreground object. And we will leave this for future improvement.

Acknowledgments

The authors would like to thank Professor Bhiksha Raj at and Teaching assistant Sourish Chaudhuri. Without Professor Bhiksha Raj careful preparation of the course, we do not have opportunity understanding such deep knowledge. The Professor and Teaching assistant also spent a lot time to help us work out many problems. Without their help this project could not reach its present form.

References

[1] Le Lu, NJ & Gregory Hager, MD. (2006) Dynamic Foreground/Background Extraction from Images and Videos using Random Patches. NIPS 2006.

[2] S.M. Al-Garni and A. A. Abdennour, “Moving Vehicle Detection using Automatic Background Extractuib,” World Academy of Science, Engineering and Technology 24,2006

[3] Zhang Yuan; Jia Kebin; Liu Pengyu; , "Video Stitch Algorithm Based on Dynamic Foreground Extraction," Image and Signal Processing, 2009. CISP '09. 2nd International Congress on , vol., no., pp.1-5, 17-19 Oct. 2009. doi: 10.1109/CISP.2009.5305636

[4] L. Li, W. Huang, I.Y.H. Gu, and Q. Tian, “Statistical Modeling of Complex Backgrounds for Foreground Object Detection”, IEEE Transaction on Image Processing, Vol. 13, No. 11, 2004.

[5] Zhang Yuan; Jia Kebin; Liu Pengyu; , "Video Stitch Algorithm Based on Dynamic Foreground Extraction," Image and Signal Processing, 2009. CISP '09. 2nd International Congress on , vol., no., pp.1-5, 17-19 Oct. 2009. doi: 10.1109/CISP.2009.5305636