

Human Detection in Surveillance Applications

Ashish Desai

Abstract

To attack the problem of detecting humans in a surveillance video, there are a few techniques that can be used. In this project, both a color based algorithm and a motion based algorithm were implemented, eventually leading to a combined approach. While each method had advantages and disadvantages, the combined approach led to an algorithm that worked exceptionally well with lateral movement relative to the camera, as well as with objects that were sufficiently far from the camera. While movement towards and away from the camera, especially very near to the camera caused some problems, I believe extensions to the project can improve upon the algorithm to make it very robust.

Introduction

When trying to determine a good project to pursue regarding digital video processing, I decided to look around in my everyday life and look for a problem to solve. During this time, the parking garage I used often had petty vandalism problems, where criminals were breaking into the garage and vandalizing cars. To prevent this, our landlord had installed cameras, but this did not deter criminals, because they knew that the viewing of the tapes would happen later when they were long gone. So, I began to wonder: what if we had a real-time count of all the people in the garage, along with their pictures?

Problem Statement

Really this problem can be easily expanded to any surveillance application. In terms of our class, this breaks down to a foreground/background segmentation problem, which can then be coupled with segmenting the resulting foreground into separate objects. Formally, given a video sequence, I would like to determine the number of people present at every frame, as well as save a picture of them.

Methodology

To attack the segmentation problem, I decided to attempt two separate approaches (and eventually I decided to combine them). The first approach was to use a color based segmentation algorithm that will look at each pixel in a frame and determine whether it is in the foreground or background based on its color. The second approach was to use a segmentation algorithm based on motion estimation to determine whether each block in a particular frame was in the foreground or background based on the next frame in the sequence. Each of these methods is described in detail below.

A 25-second video sequence was captured at 10 frames per second in my apartment where each frame contained between zero and three people. The algorithms described below were used to estimate the number of people in each frame and the results will be discussed in the subsequent section.

Color Based Segmentation

The color-based segmentation that I used was based on the Staufer and Grimson paper given in class (see references). I do not want to restate the entire algorithm, but the basis for this method is that every pixel in a frame has a probability of being a particular color based on a mixture of multiple Gaussian probability distributions. If a particular pixel is considered a part of the background, it should remain constant (within a small variance due to noise sources) for long periods of time. However, as objects move in front of a background, a particular pixel will change color briefly and then return to the original color. Therefore, if we keep track of multiple Gaussian distributions for each pixel, and determine which of these Gaussians is in the foreground and which are in the background, we can make a good estimation for our current pixel, given the various probabilities. This is done by updating the mean and variance of each Gaussian based on the current values, determining the most probable Gaussian and determining whether this Gaussian is in the foreground or background. For more details, please refer to the Staufer and Grimson paper given in the reference section.

For the most part, the algorithm was used as described in the paper, using a learning constant of $\alpha = 0.7$ and $K = 5$ Gaussians. Some notable differences between my algorithm and that of the paper is that I determined that a pixel value must be closer than 1.5 standard deviations from the mean to be considered a sample from a particular Gaussian, versus Staufer and Grimson using 2.5 standard deviations. Also, I initialized the mean and variances of the Gaussians to random values with relatively large variances, which leads to the initial frames of the sequence to be considered complete background.

Without regurgitating the Staufer and Grimson paper, each pixel is considered background if it is assigned to the Gaussian with the highest ratio of weight/variance (this effectively means it is the most recently present Gaussian with the highest probability). When simply implementing this algorithm, I found that in addition to the desired targets, many other pixels were still considered foreground. However, often they were isolated pixels, so I combined the above algorithm with a connectivity requirement that led to the elimination of any foreground pixel that was not immediately adjacent to four other classified foreground pixels.

Finally, I took advantage of the fact that I knew I was looking for fairly large objects. After the connectivity requirement, there were clusters of pixels that were very close to each other but not touching. Therefore, I connected groups that were within a 4x4 block of each other. To take advantage of the large target size, I finally eliminated any group that contained fewer than 100 pixels.

As a note, I purposely wanted to set up my color segmentation algorithm to prevent any false detection of foreground. This was done so that when I later combined this with the motion estimation (described later to have high false detection) I could weight the confidence of the color segmentation very high (this will be addressed more in the combination section). Therefore, I chose a fairly high value for the learning constant so objects will quickly be determined as background. Also, because I only had 10 fps, and I wanted to converge very quickly, the high learning rate was essential.

Motion Based Segmentation

The method for motion based segmentation was very simplistic, both because more time was spent to improve the color based segmentation, and because of the desire

to reduce complexity for the long-term goal of a real-time algorithm. Basically, for each frame in the sequence, a simple block-matching algorithm was implemented between the current frame and the next frame. For the block matching, I used 16x16 blocks with full search over a $\pm 16/-15$ search area with a SAD criterion. Then, a simple threshold criterion was chosen, in which any block that was determined to move more than 8 pixels (in any direction) was considered a foreground object. While I did look at other means to do segmentation, such as K-means clustering, I was trying to reduce complexity for the reasons stated above. Additionally, I wanted the output for the motion-based segmentation to have the property that it would not eliminate any foreground blocks by classifying them as background. This was imperative for the combination approach described below. Again, to reduce the effects of noise, I also put size constraints on any foreground objects such that any object that contained fewer than four blocks was no longer considered part of the foreground.

Combination of Color and Motion Based Segmentation

As both of the previously described methods had their benefits and disadvantages (described in the Results section), I decided to combine the two methods to provide the best detection throughout the video sequence. As described above, the parameters for the color-based segmentation were chosen such that there was a very high confidence that its output was indeed foreground pixels, but perhaps not all of them were captured. Additionally, the motion-based segmentation was developed such that all the foreground pixels were captured, but perhaps many background pixels as well. Therefore, a weighting was done with each of the outputs where more weight was given to higher connected foreground pixels, and those in the color output were weighted heavier than those in the motion-based output. Finally, a threshold was heuristically determined to provide a nice balance between the two methods.

The block diagram of the overall system is shown below in Figure 1.

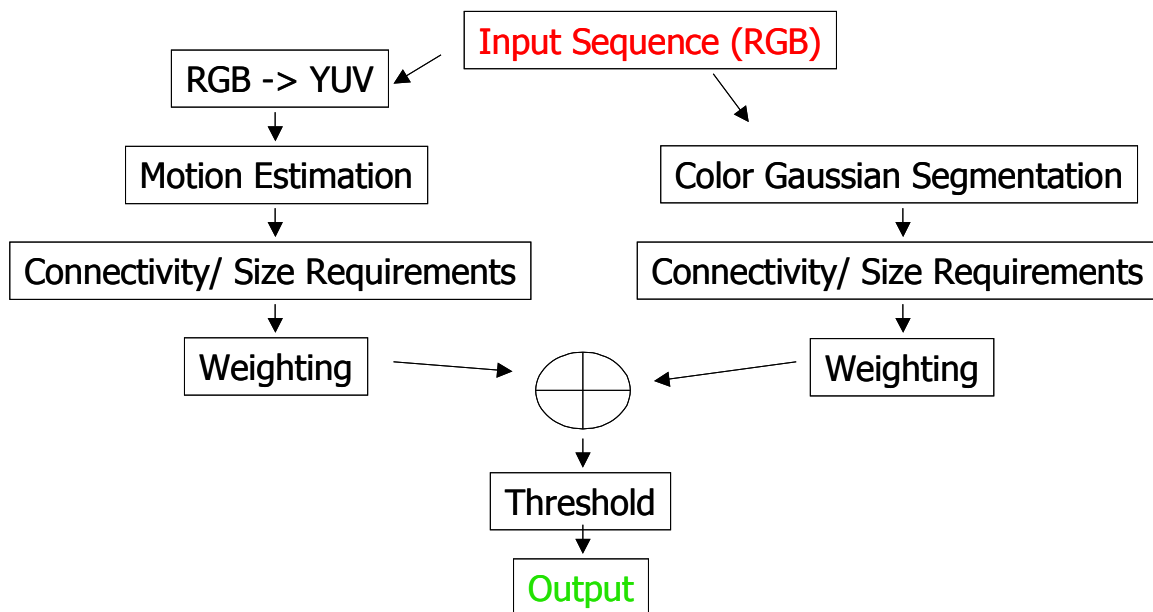


Figure 1. Block Diagram

Results

Throughout the methodology section, I alluded to some of the results of the various methods, as they led to modifications of some of the algorithms. Overall, the color based segmentation performed very well (I spent the most time adjusting the parameters of this method), while the motion based segmentation had an adequate performance.

More specifically, the color based segmentation mainly had problems when there were large occlusions for a number frames, and then they disappeared. This is probably because the learning constant was chosen to be fairly high, so these large occlusions caused the previous background to be lost from memory. Also, the color based segmentation had problems with shadows being considered a foreground object. Overall, however, most of the output of the color based method was in fact foreground (but not necessarily all of the foreground).

On the other hand, the motion based segmentation was designed to identify all foreground objects, with the trade-off of falsely identifying background as foreground. This method successfully did this, where the falsely detected background occurred mostly from reflective surfaces (possibly detected moving objects out of camera view) and the walls near the camera (possibly from automatic gain control of the camera when foreground objects moved towards or away from camera). Additionally, the motion estimation had some problems with aspect ratio changes, because this was based on block matching, rather than affine parameters. Finally, there were a few frames that underwent global motion by a person bumping the camera. This was not handled well by the motion based segmentation, because of the use of a simple threshold versus clustering of motion vectors.

Finally, the combination of these methods actually worked quite well. For lateral movement only, the detection worked exceptionally well, whereas with large occlusions or aspect ratio changes still posed problems. Again, the combination performed better than the individual algorithms. Specifically, the problems with shadows and global motion were completely eliminated.

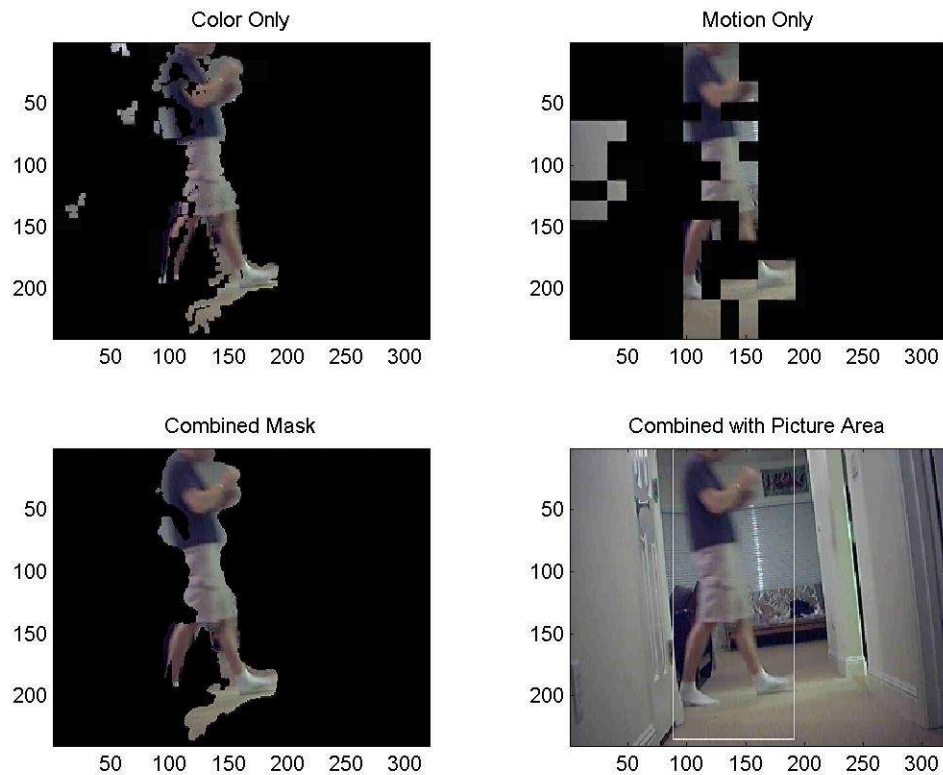


Figure 2. Comparison of Methods

Figure 2 shows the output of the various methods, where a black pixel value means that it was considered background. Here we see that the color based segmentation was pretty accurate, the motion based identified the object, as well as some of the wall, and the combined method did an excellent job. The picture with the box around the foreground object is the final output given by the algorithm to provide the goal of counting the number of people and providing a picture. Note that this example frame involved only lateral motion.

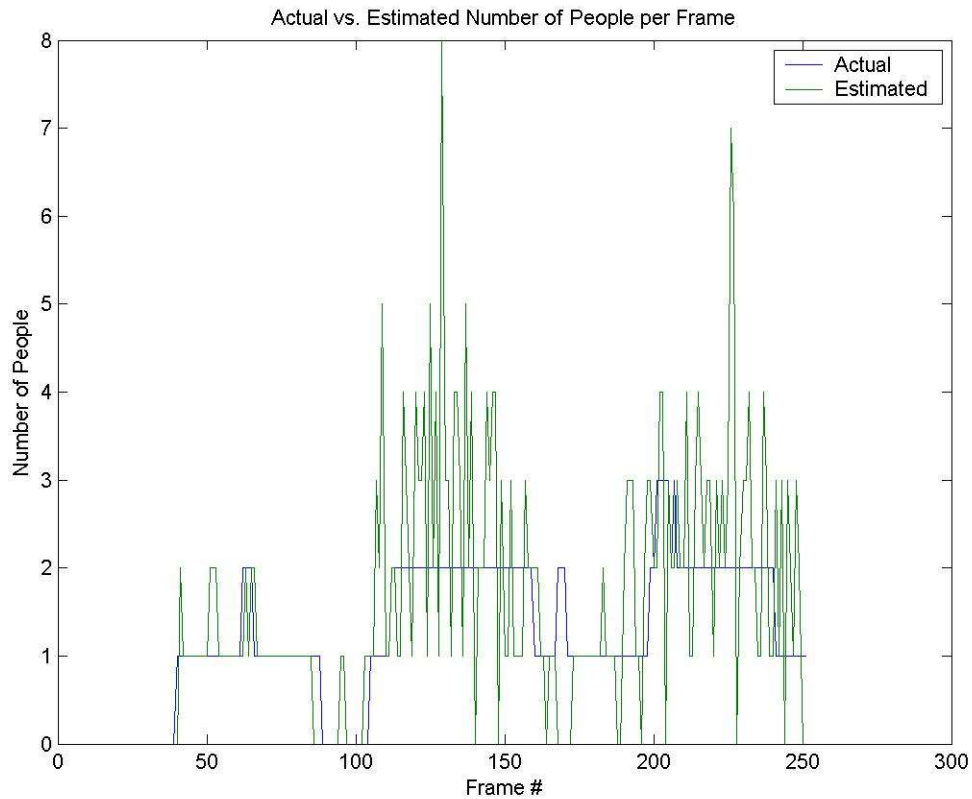


Figure 3. Number of People per Frame

Figure 3 shows the actual number of people in each frame, as well as the estimated number given from the combined algorithm. From frames 0 through 100, there was either no motion, or only lateral motion. As was mentioned earlier, the algorithm works exceptionally well (mean error of 0.13 people per frame) with this type of motion. Frames 100 through 175, as well as Frames 195 through 251 include very large occlusions, as well as large changes in the aspect ratio of objects. Both because of lack of affine parameters and automatic gain control of the camera, the algorithm contains many errors during this period of time (mean error of 0.99 people per frame).



Figure 4. Lateral Motion Frame

Figure 4 shows a frame where only lateral motion occurs, and the algorithm works very well.

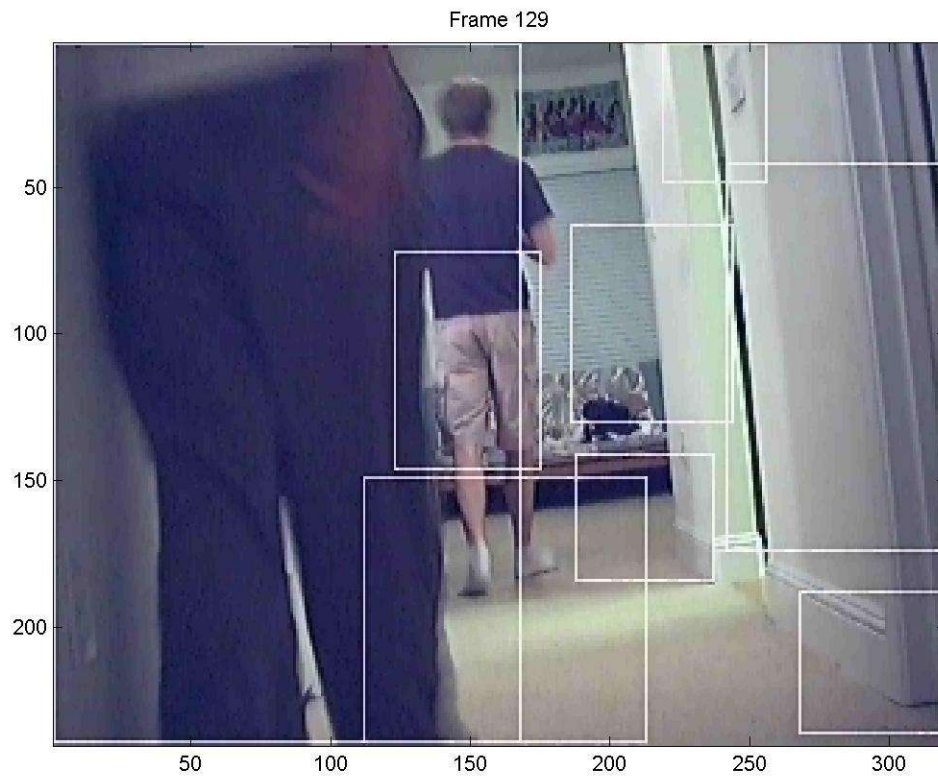


Figure 5. Frame with Occlusion/Aspect Ratio Changes

Figure 5 shows a frame with changes in aspect ratio, occlusion, and possible automatic gain control changes (object suddenly appears immediately in front of camera). Notice the large number of false detections, as well as the missing detection of the large foreground object that just appeared in front of the camera (on the left).

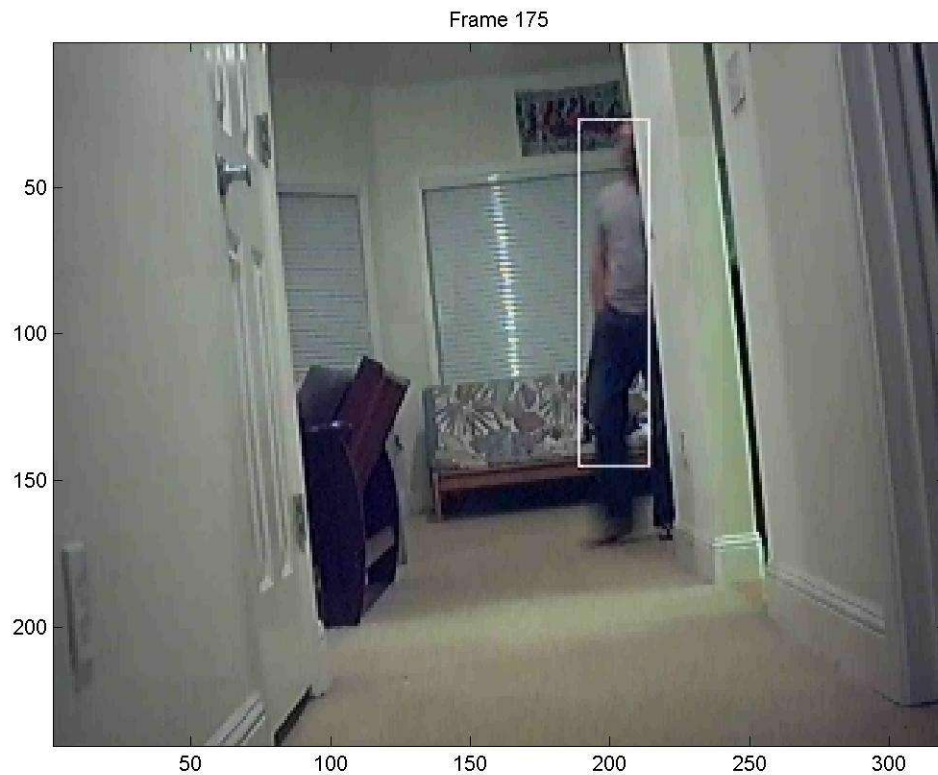


Figure 6. Aspect Ratio Change Far from Camera

Figure 6 shows that the change in aspect ratio was detected properly as long as the object remained further away from the camera. Again, this may be due to the fact that the automatic gain control of the camera is not initiated, and the color based segmentation is weighted more heavily than the motion based segmentation.

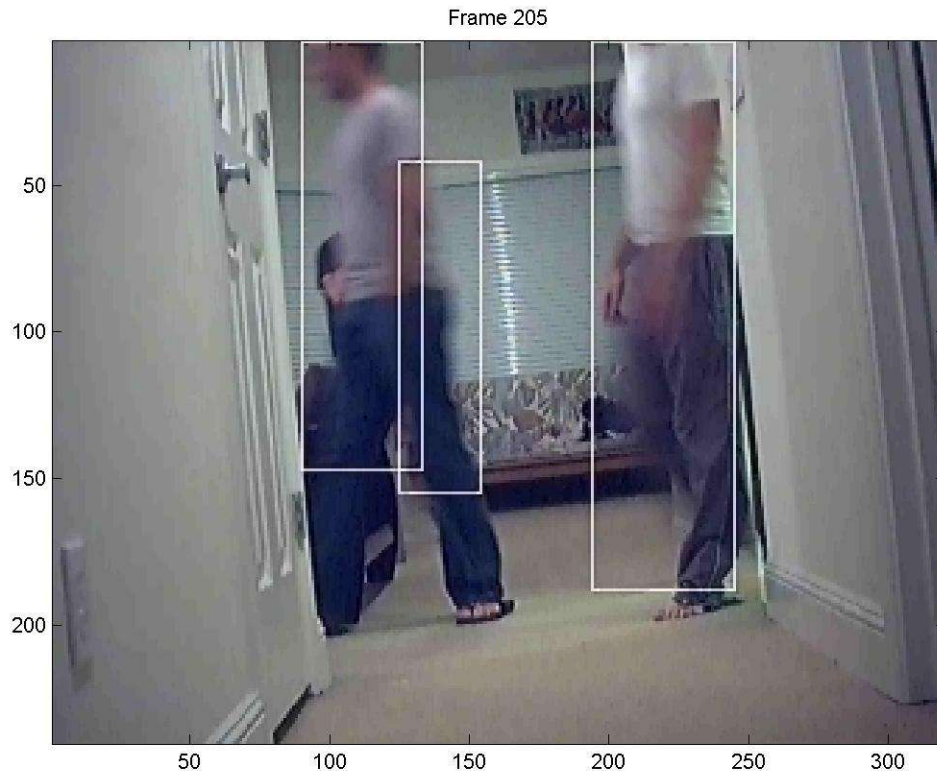


Figure 7. Connected Foreground Objects

Figure 7 shows an instance where the algorithm tries to identify two connected people (on the left), but cannot accurately separate them. This is due to the fact that the only connectivity requirements and thresholds were used. A possible improvement upon this method would be to use the motion vectors to segment two foreground objects that may be connected based on other algorithms.

Conclusion

Overall, I believe that this project was very successful in achieving the goal of identifying the number of people in a frame as well as providing a picture of them. While there is always room for improvement, the algorithm works exceptionally well for lateral movement relative to the camera, and for objects that are not close enough to the camera to cause large occlusions of the background.

To improve upon this algorithm, many other methods could be considered. For example, the use of affine parameters for motion-based segmentation would improve the problems with large aspect ratio changes. The use of a log-based search for block matching could provide an opportunity to make the algorithm truly real-time (I expect that this could have been achieved for the color based segmentation). If K-means clustering was used for motion-based segmentation, this would better handle global motion, as well as perhaps segment multiple people that are connected from the color algorithm. Finally, the implementation of a temporal constraint that would utilize the knowledge of detected humans from previous and/or future frames could definitely improve upon this algorithm.

References

1. Chris Stauffer and W.E.L. Grimson. “Adaptive background mixture models for real-time tracking”, *CVPR99*, Fort Collins, CO, (June 1999).