STATIC-FRAME SMOOTH OBJECT TRACKING

Koger Darden
EE 371R - Bovik

1. ABSTRACT

The project is a solution for smoothly tracking objects in a static-frame video (i.e., a constant background image). This concept can be used to actively zoom-in an object of interest, as well as reduce video storage size by removing uninteresting parts of a video. Two methods were used to approach this; the first method involved calculating the difference between consecutive frames, and the second method involved extracting a foreground video. Various filters were used to reduce noise in each of these approaches, including: gaussian, median, and binary thresholding. Following filtering, activity centers were calculated in order to detect where the object was located. Hopefully, this paper will give the reader an understanding of which methods prove effective in tracking objects smoothly so that focused videos can be extracted.

2. PURPOSE

Often times in static-frame videos the object of interest only takes up a small portion of the frame, leaving a large amount of unused space. This can lead to difficulty seeing the object of interest since it is very zoomed-out, as well as resulting in storage inefficiencies. If the object of interest can be located within the video, a sub-video can be extracted, resulting in a zoomed-in object of interest as well as a smaller video storage size.

This can be applied in many applications. Security cameras, for example, can use this to more effectively call attention to a particular subject of interest. Furthermore, it would allow the camera to only save the portion of the image that has activity occurring within, allowing for more video time to fit on a particular storage device. Another application is for simple viewing of an image, allowing a viewer to have an easier time observing a particular object of interest rather than having to follow it around the frame.

3. METHODS

I used two main approaches for this project. The first involved calculating the differences between adjacent frames, and the second involved extracting a foreground video seperated from a background image.

3.1. Frame difference

For the first method, I first converted to grayscale, then calculated the pixel-wise difference between adjacent frames to capture movement in a “difference-video”. Each frame of this difference-video was then averaged to find a single activity-center for that frame by multiplying each pixel value by its row and column index, summing the products for these independently (i.e., one sum for row, one for column), and then dividing each sum by the sum of all pixel values for the frame. This process produced a single (r,c) coordinate for each frame representing the centerpoint of where a tracked object was located.

In order to reduce noise in the aforementioned method, the difference-video was passed through various filters before calculating the activity-centers. Three filters were used: a median filter with a 6x3 window size, a gaussian filter with a with a sigma value of 3, and a binary thresholding filter thresholding at 20%. Each filter was applied separately to the original difference-video to find which resulted in the best object tracking.

3.2. Foreground extraction

The second approach involved converting to grayscale and then calculating a background image for the entire video. A foreground video was then found by subtracting this background image from each frame. The background image was found by calculating the mode average value for each frame across the entire video duration. The intuition behind this is that a moving object will only briefly visit each pixel on the screen and thus will not influence the mode. Activity-centers for each frame were then calculated using the foreground video.

Similarly to the first approach, the foreground video was filtered to reduce noise before calculating activity-centers. However, this time only a binary thresholding filter thresholded at 20% was used. The result of this was compared to the object tracking obtained by using the unfiltered foreground video as well as the results from the difference-video and associated filtering.

Following this, to smooth out the object tracking even further, the activity-centers of neighboring frames were averaged to reduce jittering. For this, one pass over the video frames calculated all of the original activity-centers, and then a second pass averaged these values using a window. This was only applied to whichever method described previously performed best.

Finally, the resulting averaged activity-centers were used to extract a new video with a manually specified window size that stays focused on the object of importance.

4. RESULTS

From the first approach, the least-effective result was obtained without using filtering. The object tracking jumped all around the video only very generally moving with the object across the screen. The median filtered difference image resulted in better results, tracking the moving object jerkily and still jumping away occasionally. Using the gaussian filter had similar but slightly better results with slightly less jumping away. All of these object tracking methods had a trend of being pulled towards the center of the video due to the average noise of the image. However, The binary thresholding filter performed rather well, tracking smoothly without jumping, with the exception of losing track of the object when movement ceased.

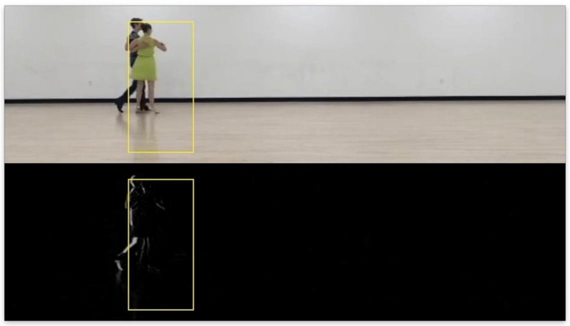

Fig. 1: Difference video

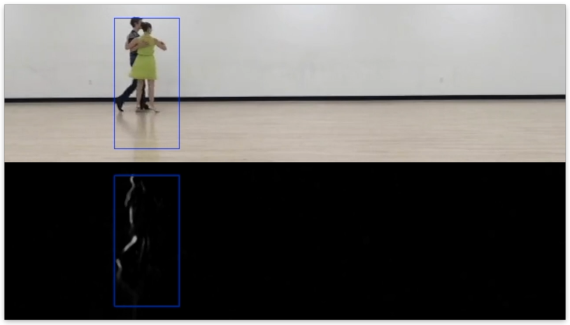

Fig. 2: Median filtered difference video

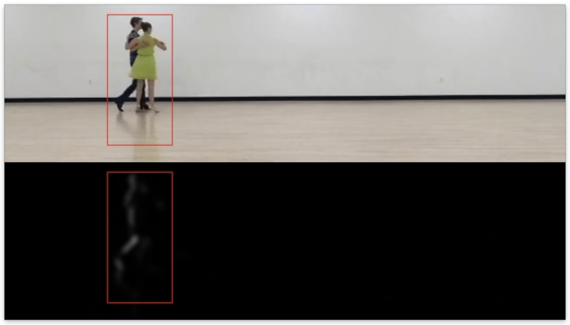

Fig. 3: Gaussian filtered difference video

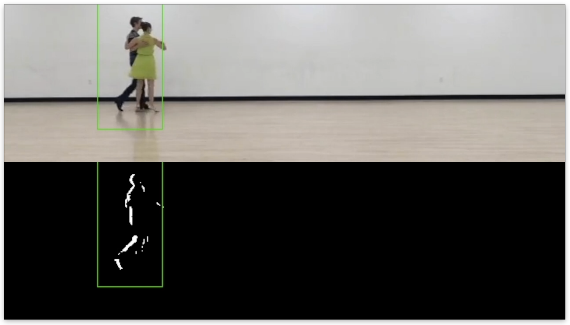

Fig. 4: Binary thresholded difference video

The second approach produced much better results as much more information was present in each frame. Also, since this approach was not based on the frame-to-frame difference, it was able to track the object even when no movement was present. Without filtering the result was smooth, but the tracking was again pulled towards the center of the image due to noise. Using the binary thresholding filter solved this problem and resulted in a smooth tracking that wasn't influenced by noise.

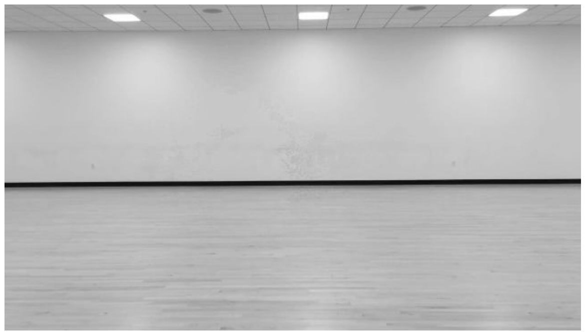

Fig. 5: Extracted background image

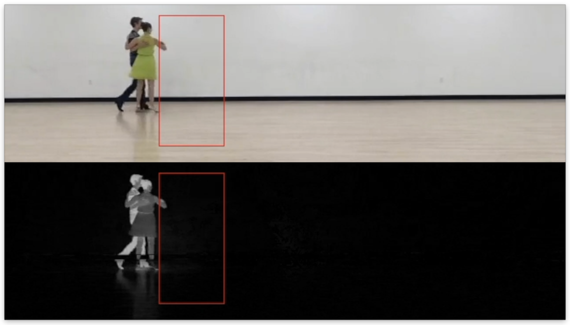

Fig. 6: Foreground video

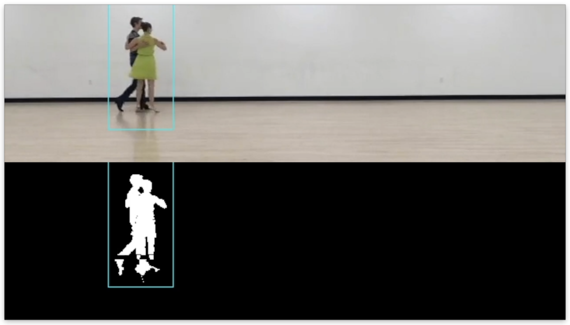

Fig. 7: Thresholded foreground video

The binary filtered foreground approach was then used in the frame-to-frame activity-center window averaging. Through trial and error, a window size of 25 proved to be most effective. This further smoothed out the image, substantially reducing the small tremors in object tracking present due to the differences between frames.

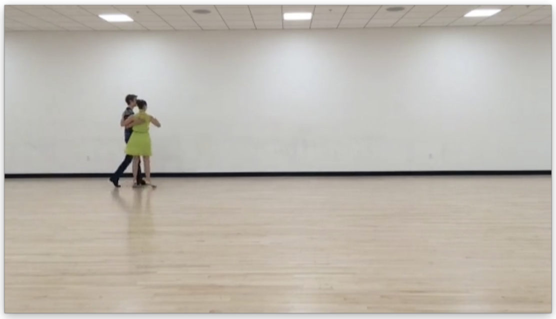

Fig. 8: Original video

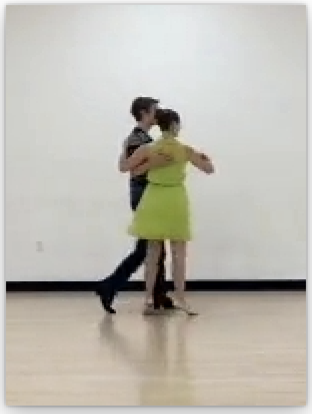

Fig. 9: Extracted object tracking video

Following this process, the video of the tracked object was exported using a manually selected window size. In this newly produced video, the object stayed focused directly in the center the entire time.

5. WHAT I LEARNED

This project taught the basics of motion tracking by using the differences found between images. It was interesting to see how a background image that is never fully present can be calculated from a video. It also showed me just how much unseen noise can be present in a seemingly empty space, and how it can compound to cause negative effects. Furthermore, it is incredibly important to use good filters, and binary-thresholding filters are extremely good at filtering at noise when the actual details of an image are not critical. It was interesting to see how different methods within each frame as well as between seperate frames can be used to smooth-out object tracking.