VIDI Surveillance

VAST 2009 Challenge
Challenge 3 - Video Analysis

1 Authors and Affiliations:

Anna Tikhonova, University of California at Davis, atikhono@ucdavis.edu
James Shearer, University of California at Davis, jjshearer@ucdavis.edu
Michael Ogawa, University of California at Davis
Chad Jones, University of California at Davis
Kwan-Liu Ma, University of California at Davis

2 Tools:

OpenCV - an open source computer vision library originally developed by Intel.
Link: `http://ubaa.net/shared/processing/opencv`
Processing - an open source programming language for developing ideas.
Link: `http://processing.org`
MATLAB - a high-level language and interactive environment for performing intensive computational tasks.
Link: `http://www.mathworks.com/products/matlab`

3 Video:

`http://vis.cs.ucdavis.edu/~shearer/Video.mov`

4 Answers:

4.1 MC3.1: Provide a tab-delimitated table containing the location, start time and duration of the events identified above.

`Video.txt`

4.2 MC3.2: Identify any events of potential counterintelligence/espionage interest in the video. Provide a Detailed Answer, including a description of any activities, and why the event is of interest.

To begin looking for meetings in the security camera videos, we first filter out noise due to camera panning. We automatically separate the video frames into four separate streams, depending on camera viewpoint. We use the following heuristic for detecting camera movement: if 30% of frame's pixels change in reference to the previous frame, we mark that frame as the end of a current video segment. Then, we skip the frames in a camera pan, looking for the beginning of the next video segment. The camera doesn't come to a stop abruptly; the movement is jittery. Thus, we use a buffer of 20 frames to make sure we detect a complete stop. If the camera stops moving for 20 frames, we mark the next frame as the beginning of a new segment; otherwise, we keep looking.

Once we have four separate streams, we use OpenCV's blob tracking methods to detect and track moving objects. We set object size constraints to avoid tracking noise speckles in the videos. We store the following information for each object: position and size for every frame it is on the screen.

The list of tracked objects obtained by this method is quite large. We filter the object list based on size, aspect ratio, position on the screen, and duration on the screen. This substantially reduces the size of the object list.

Our first attempt at looking for meetings involved plotting movement "curves" for every object. We plot the movement in x and y directions separately, with x axis representing time/video frames. An example of plot is shown in the video. The top half of the image is time vs. change in x position and bottom half - time vs. change in y position. We hypothesized that if two curves (representing two objects) intersect in both x and y directions (forming n "x" pattern), that represents a meeting. What we like about this approach is how easy it is to look for patterns (especially using a user-draggable line shown in Figure 1a). We realize that the objects do not necessarily have to come in direct contact for the meeting to occur. They can, instead, come within speaking range of each other. Thus, we considered computing the proximity of each curve to each neighboring curve. That would allow us to visually show two objects coming in contact with each other in a visualization. For example, we could color the area between curves in red; the closer the curves are to each other, the brighter the red color. Since the area between the curves could be small, we considered using an overlay or a simple abstract view with colored regions, representing potential meetings.

The problem with the above approach is that plotting only 5 minutes of security video this way would take up the entire screen space and even the abstract view containing only the red regions would still include all the empty space in which no meetings occur. We realized that since we are looking for meetings between two people, our analysis as well as our visualization should be based on the analysis of objects.

Examining the movement curves, we hypothesized we should be looking for the following pattern: an object moves for a certain period of time, stops (for a meeting), and then walks away. These events correspond to infection points on the movement curves. We first used MATLAB to reconstruct and simplify movement curves. Then, we found inflection points using a simple algorithm that finds points that are higher/lower then their two nearest neighbors. The inflection points correspond to frame numbers at which objects appear and leave the screen, stop, or begin movement. However, they did not provide us enough information and thus we sampled the points instead.

We use those representative frames to create a single image for each object of interest. That image fully demonstrates object's activities in the video. In order to create this image, we take the video frame that corresponds to the first sample point and use it as background. Then, we take frames that correspond to the rest of the sample points, extract the object using a foreground mask (computed from the position and size of the object - output of the blob tracking algorithm), and composite the masked object over the background frame. An example of such an image is shown in the video. This operation is performed for every object. We also dilate and blur the foreground masks a little to capture the second object involved in a potential meeting.

Once we have the representative images for each object, we combine them into a single explorable visualization. The user may look at the images one at a time and mark each as a meeting or not a meeting.

We looked through all the images and manually selected the images that we thought corresponded to meetings. We found the following events particularly suspicious:

There is a briefcase exchange between a lady in a hat and a beige trench coat and a man in a black suit.
Location2 3:23:52 06:11

Also, there are two meetings between the same lady and other men.
Location4 2:00:53 05:16 and Location4 2:32:01 02:33

These events (along with a few additional ones that we found suspicious) were easily identified by our system.