Assignment 2: Tracking Surveillance Based on Computer Vision, Fall 2015

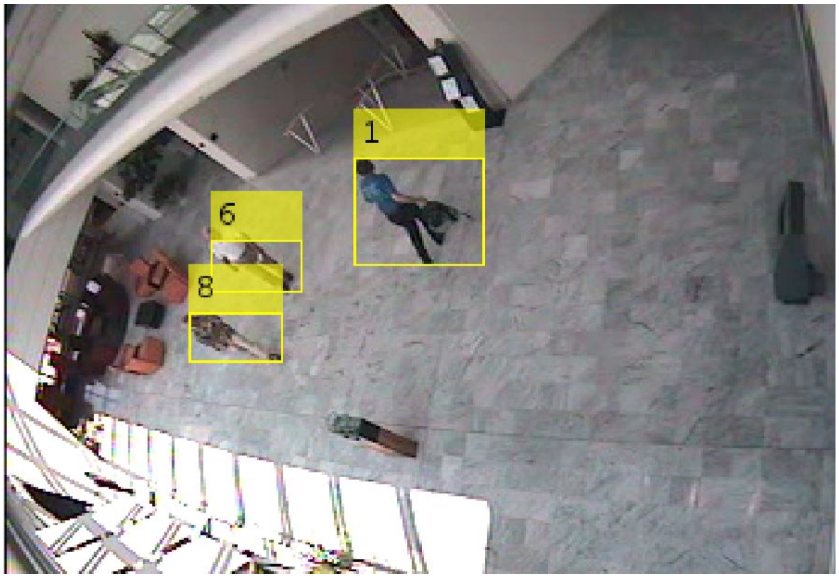
Ofer Berkovich - 039860390
Itay Radotzki - 204647366

** Note that instead of bounding colors we are using ids (numbers).*
In addition, we borrow some practical ideas from a Matlab article about Motion-Based Multiple Object Tracking: http://bit.ly/1UXK5xw

Pointers:
1 => naive-tracking
2 => motion-prediction-tracking
4 => histogram-motion-prediction-tracking
5 => histogram-tracking
6 => accident-alert

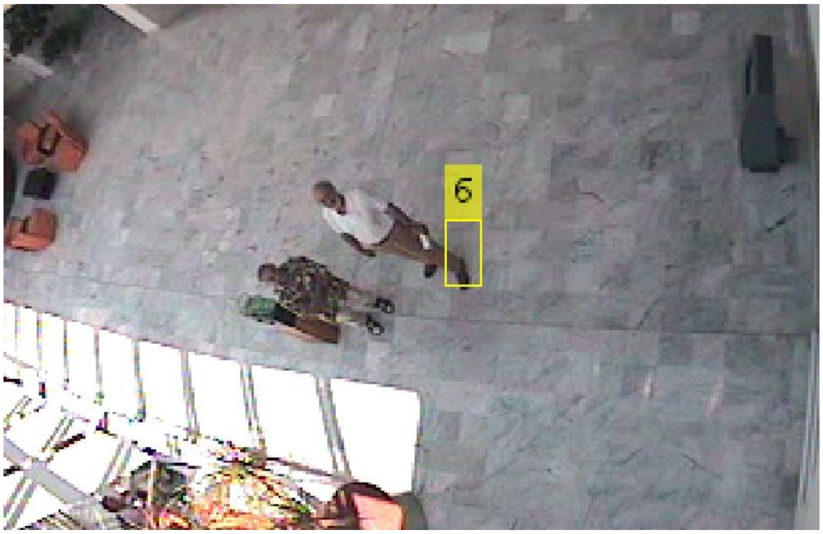
Each of the programs has a README.md file with instructions.

1, e. The naive tracking works pretty well. As you can see below we can detect multiple objects in the video, even when they are close to each other.



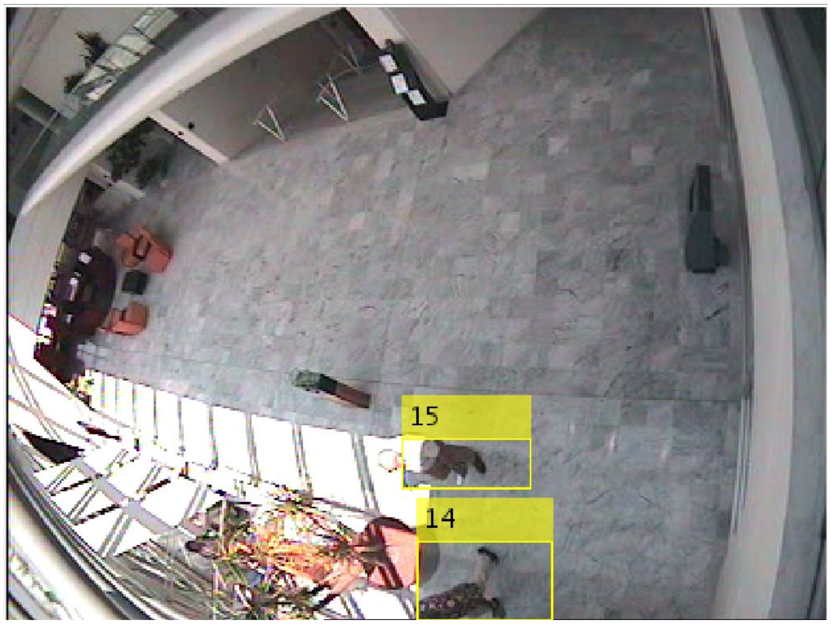
The drawbacks are:

- When the boundary boxes of some pedestrians overlapped we detect them as a single entity. When they will move away from each other and their boundary boxes will not overlapped, they will be detect as unique entities. This scenario can cause a fault id increments (like you can see in the image above - pedestrian 6 and 8 should be marked as 2 and 3)
- When a pedestrian stays still, the background subtraction does not mark him and that cause false positives detections, see image below.



2, a. In generally pedestrian’s blob size expected to change over the frames because the input video is 2D (no depth) - so as the pedestrian will come toward the camera he will look bigger (represented with more pixels in the frame).

2, d. The motion prediction tracking is also works pretty well:



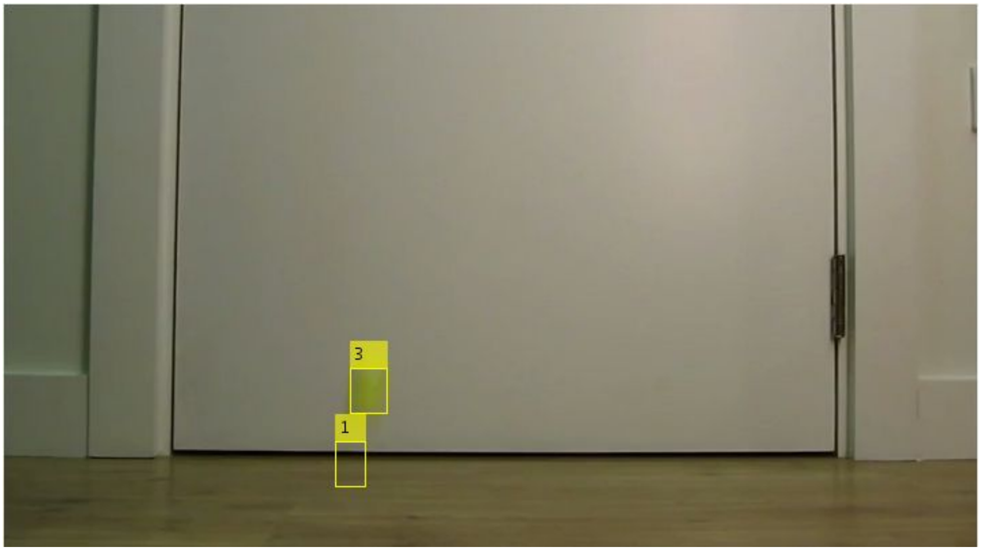
Also here we have the same issue with stand still pedestrian (When a pedestrian stays still, the background subtraction does not mark him and that cause false positives detections).

In our case it did not improve the result from the last naive implementation. It expected to improve the results on that scenario: two pedestrian walking toward each other until a meeting point, and then continue to move. With the motion prediction we will be able to predict the movement of each of the pedestrian and successfully detect them after the meeting point.

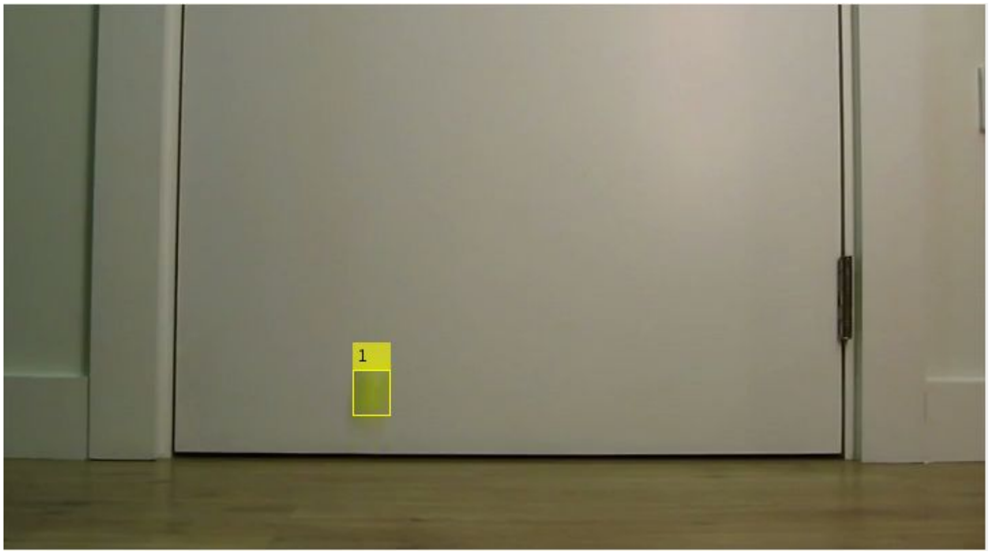
3. By predicting the next position of the target, the change detection algorithem can preper it self for it by marking the area in which the target is predicted to be at and look for changes compared to the the last frame before the target entered, for example giving more weight to the last frame in the background matrix. Now if the target should enter as predicted the change compared to to last frame will be greater and the blob will be clearer. leading to a better tracking.

4, c. when a target changes direction suddenly, for example tracking a bouncing ball - the prediction will be in the motion direction of the ball but once the ball hits the ground the prediction won't find it.

Here is the result when running with motion prediction:

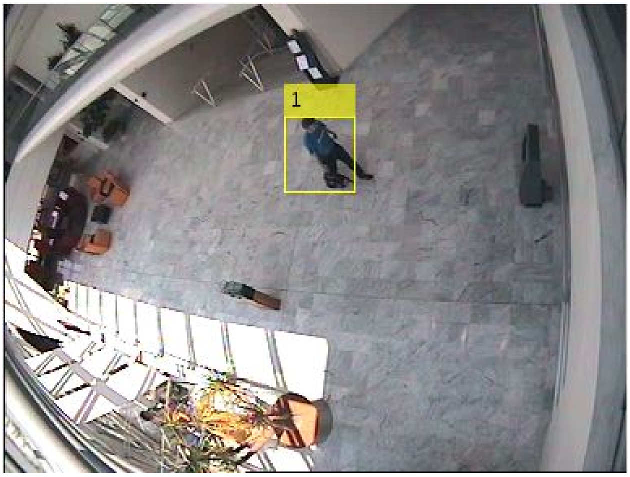


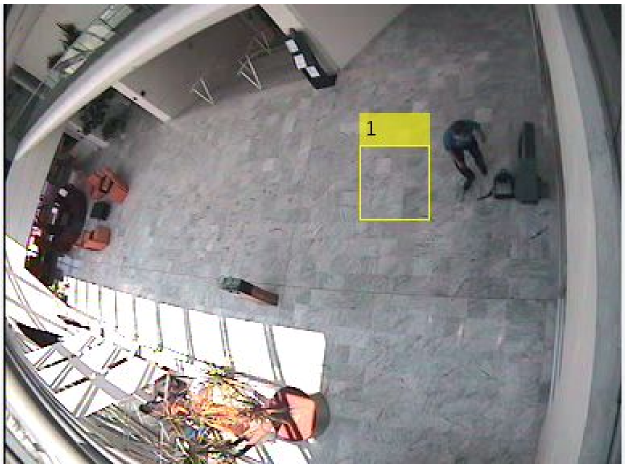
And here is the result when running with 'histogram' on 'bhattacharyya' distance:



5, a. First we initialized objects locations. Then, for each frame in the video we create integral histogram and check if we need to create a new track at this frame (according to the initialized objects locations variable). The integral histogram is a 4D matrix; for each pixel in the frame we store 3 vectors that represent the integral histogram for that pixel (sum of all the histograms above and to the left of (*x*, *y*)). When we need to create a new track we use the integral histogram that we already calculated in order to extract the 3-channels histogram for the object's region. After doing so, we perform an exhaustive search in a region around the predicted location of each of the tracked objects. The predicted location of the object in our case is the exact place where the object was in the last frame. We extract the 3-channels histogram for each region we come across in the exhaustive search, and compare that histogram with object's histogram, using one of the available histogram-different-function (SSD, Angle, bhattacharyya). Then we update the object's location with the region that his histogram had the minimum distance from the object's histogram.

5, b. In the sequence below we can see that the tracking succeeds in following the person when he enters the room and leaves it, but when he leaves the bag on the ground the track hold the floor to his left - this is probably because the colors on the right are darker compared to the tracks histogram, this could be fixed by updating the histogram.





5, c. In the first algorithm (2) we use the change detection and look only at the motion prediction of the object. The problems with this was that if the target changed direction we lost the track and made a new one, and if it stood then the blob catched by the change detection algorithm was inconsistent which appeared as if the target is moving in different directions.
The second algorithm (4) looked for similarities in the histogram of the blob from the change detection, this sometimes led to mistakes due to similar targets or change in the appearance of our target. By involving the the motion prediction, we eliminated the blobs that were too far from the prediction.
The third algorithm (5) used only histograms, and we had to initiate the track for each object. For each frame we built an integral histogram map, and used it to search for similar histograms in the area of the last blob.

6, a. Use the motion prediction tracking, and extend it to predict N future frames. If in one the predicted future frame you see an overlapped boundary boxes between different objects give an alert.
This method will probably fails when two pedestrian will walk toward each other and then stop or change their direction (false alert - false negative).