CS-763 PROJECT REPORT

RUDRAJIT DAS 140020012

ADITYA GOLATKAR 14B030009

REFERENCE PAPER:

Real-Time Tracking of Non-Rigid Objects using Mean Shift by
Dorin Comaniciu, Visvanathan Ramesh, Peter Meer
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.28.41&rep=rep1&type=pdf

AIM:

Our objective is to explore **real-time tracking of non-rigid objects** based on **visual features such as color and/or texture**, whose statistical distributions characterize the object of interest using **mean shift iterations**. This method is appropriate for a large variety of objects with different color/texture patterns, being **robust to partial occlusions, clutter, rotation in depth, and changes in camera position**. The mean shift iterations are employed to find the target candidate that is the most similar to a given target model, with the similarity being expressed by a metric based on the **Bhattacharyya coefficient**. This method also has **low computational complexity**.

ALGORITHM AND SOME DETAILS:

Given the distribution $\{\hat{q}_u\}_{u=1\ldots m}$ of the target model and the estimated location $\hat{\mathbf{y}}_0$ of the target in the previous frame:

1. Initialize the location of the target in the current frame with $\hat{\mathbf{y}}_0$, compute the distribution $\{\hat{p}_u(\hat{\mathbf{y}}_0)\}_{u=1\ldots m}$, and evaluate

$$\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{y}}_0), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} \ .$$

2. Derive the weights $\{w_i\}_{i=1\ldots n_h}$ according to

$$w_i = \sum_{u=1}^{m} \delta\left[b(\mathbf{x}_i) - u\right] \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}} .$$

3. Based on the mean shift vector, derive the new location of the target (14)

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \right\|^2\right)} . \qquad (26)$$

Update $\{\hat{p}_u(\hat{\mathbf{y}}_1)\}_{u=1\ldots m}$, and evaluate

$$\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{y}}_1), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{y}}_1)\hat{q}_u} \ .$$

4. While $\rho\left[\hat{\mathbf{p}}(\hat{\mathbf{y}}_1), \hat{\mathbf{q}}\right] < \rho\left[\hat{\mathbf{p}}(\hat{\mathbf{y}}_0), \hat{\mathbf{q}}\right]$
   Do $\hat{\mathbf{y}}_1 \leftarrow \frac{1}{2}\left(\hat{\mathbf{y}}_0 + \hat{\mathbf{y}}_1\right)$.

5. If $\|\hat{\mathbf{y}}_1 - \hat{\mathbf{y}}_0\| < \epsilon$ Stop.
   Otherwise Set $\hat{\mathbf{y}}_0 \leftarrow \hat{\mathbf{y}}_1$ and go to Step 1.

Our aim is to **reduce the "distance"** between the 2 discrete distributions **q (from the m-bin histogram of the target model)** and **p(y) (from the m-bin histogram of the target candidate)**.

The notion of “distance” is quantified as:

$$d(\mathbf{y}) = \sqrt{1 - \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right]} \ .$$

$$\hat{\rho}(\mathbf{y}) \equiv \rho\left[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}\right] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\mathbf{y})\hat{q}_u}.$$

where $\rho[p(y), q]$ is the **Bhattacharyya coefficient** of the distribution of the target candidate at location y with respect to the distribution of the target model. It can be clearly seen that when both distributions are exactly identical, the Bhattacharyya coefficient is exactly equal to 1 and the distance between them is 0. This is a result of the Cauchy-Schwarz inequality.

The minimization of this distance is equivalent to maximization of the Bhattacharyya coefficient, which can be done efficiently based on mean shift iterations that have been described in the algorithm above.
The assumption made in this algorithm is that the **distribution of the current target candidate, p(y) is not very different from the distribution of the initial target candidate, p(y0)** which allows us to perform the **Taylor Series expansion** of the Bhattacharyya coefficient of p(y) wrt q in terms of the Bhattacharyya coefficient of p(y0) wrt q **only up to the first term**. When this approximation is valid, we can use mean shift to maximize the Bhattacharyya coefficient of p(y) wrt q.

Also the weights as defined above, **require** $\widehat{p}\, u(\widehat{y}\, 0)$ **to be non-zero**.

SOME RESULTS:

1. TABLE TENNIS SEQUENCE-

FRAME 1

FRAME 40

FRAME 55(Notice Scale Invariance)

FRAME 60(Notice Occlusion Resistance)

2. BOXING SEQUENCE-

FRAME 2

FRAME 40

FRAME 75

FRAME 388(From a different shot)

FRAME 420

FRAME 440

3. FOOTBALL SEQUENCE(Poor Quality Video)-

FRAME 1

FRAME 60

FRAME 100

FRAME 150

FRAME 200

FRAME 280

4. SHOPPING CENTRE SEQUENCE 1-

In this video, the size of the object (i.e. the man) being tracked is increasing with the frames. The tracker however, is able to correctly track the man across all frames. The size of the window around the man is the same in all the frames and equal to that in the first frame.

FRAME 13

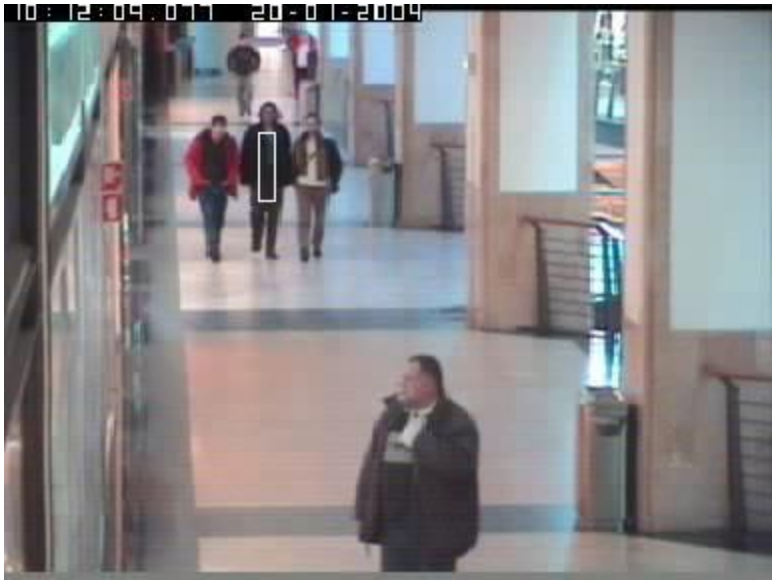
FRAME 103

FRAME 203

FRAME 303

FRAME 403

FRAME 503



FRAME 603

FRAME 703

5. SHOPPING CENTRE SEQUENCE 2-

FRAME 1

FRAME 35

FRAME 70

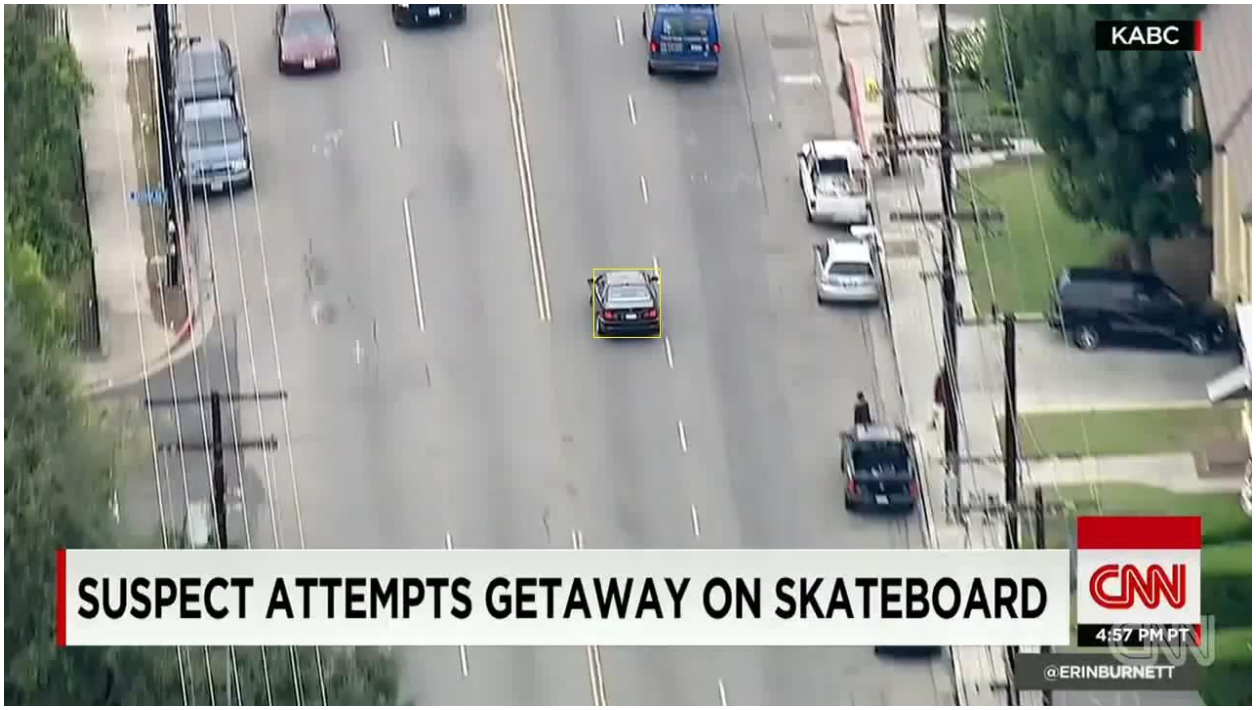
FRAME 105

FRAME 140

FRAME 175

6. CAR TRACKING SEQUENCE-

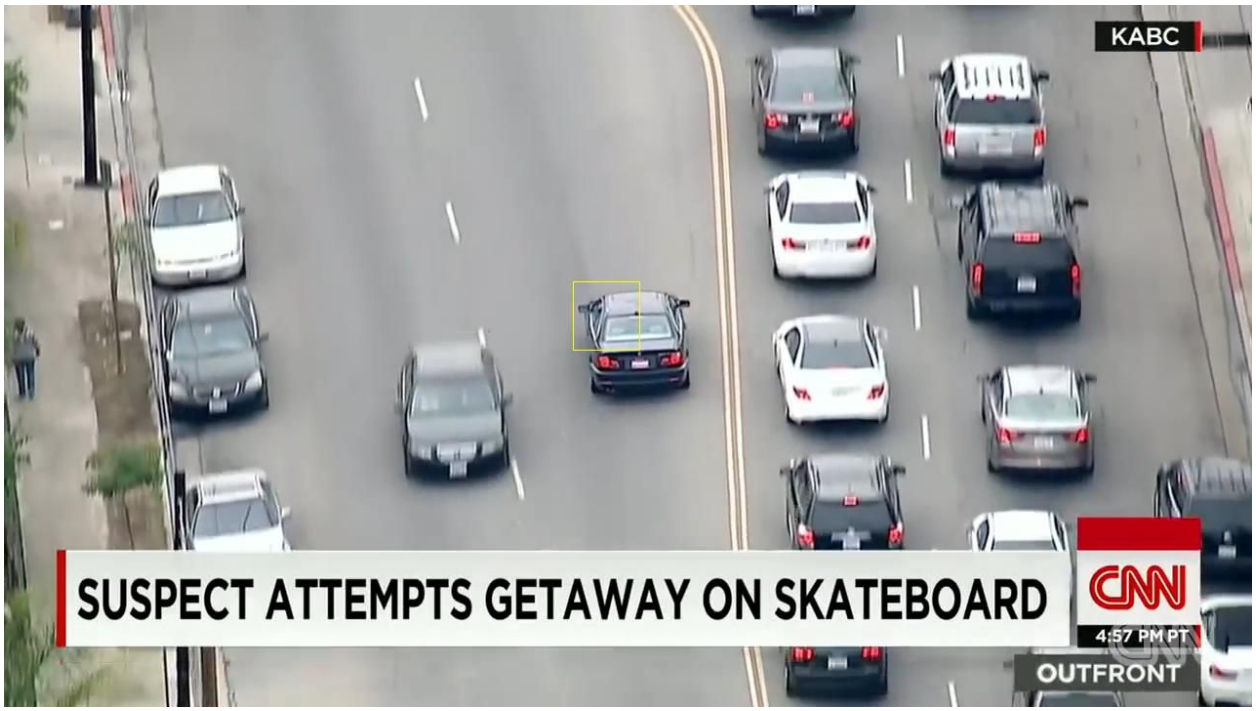
FRAME 1



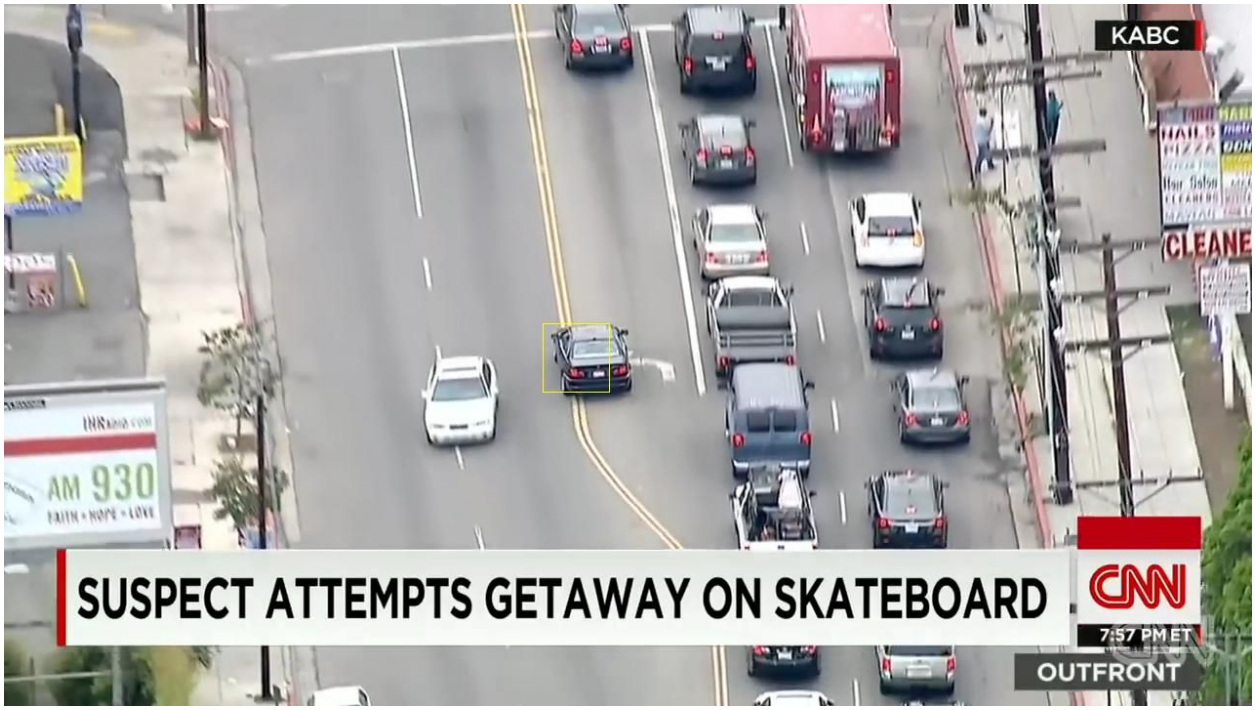
FRAME 45

FRAME 90

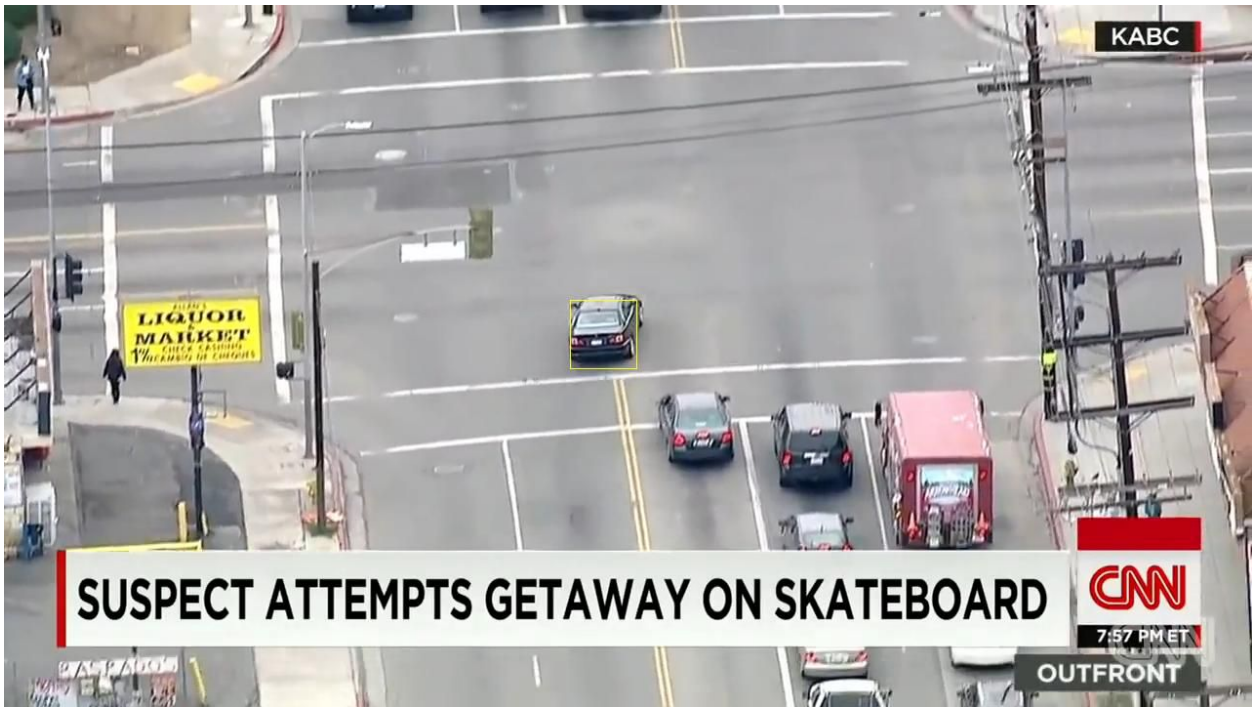
FRAME 135



FRAME 180



FRAME 235



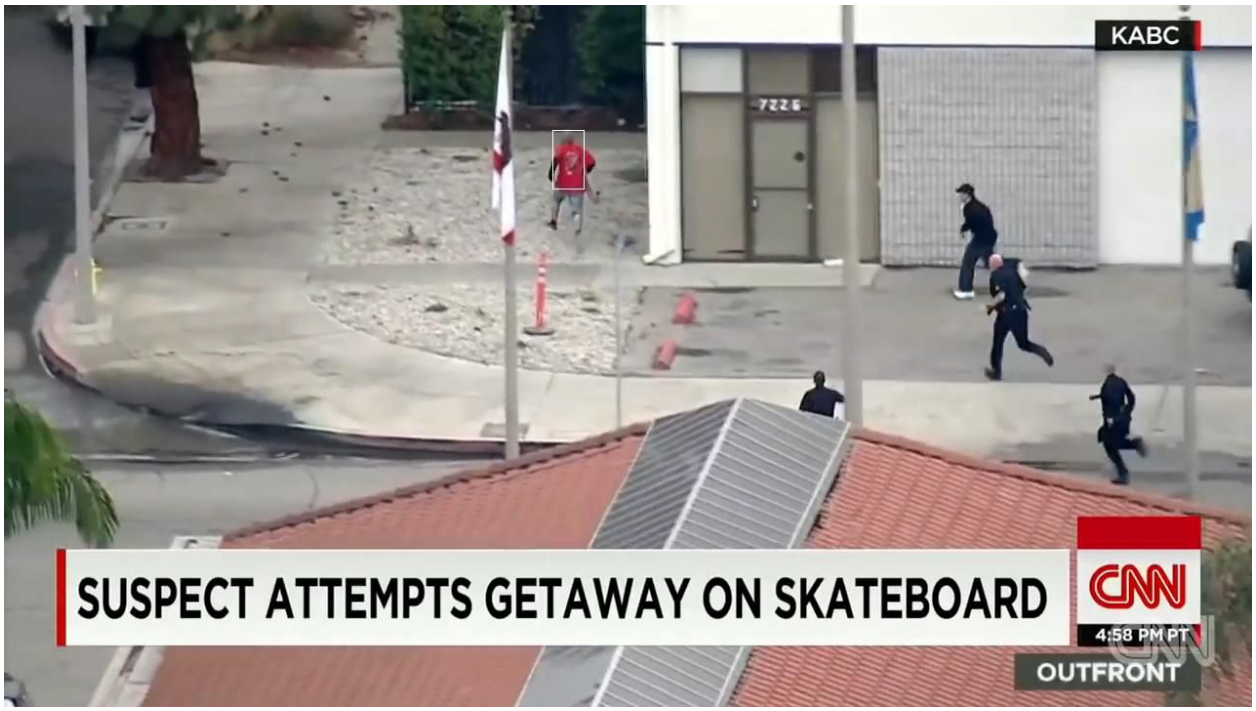
7. COPS CHASING MAN-

FRAME 1

FRAME 25

FRAME 50

FRAME 75



FRAME 100

FRAME 125

8. MAN CYCLING IN TRAFFIC-

FRAME 1

FRAME 50

FRAME 100

FRAME 150

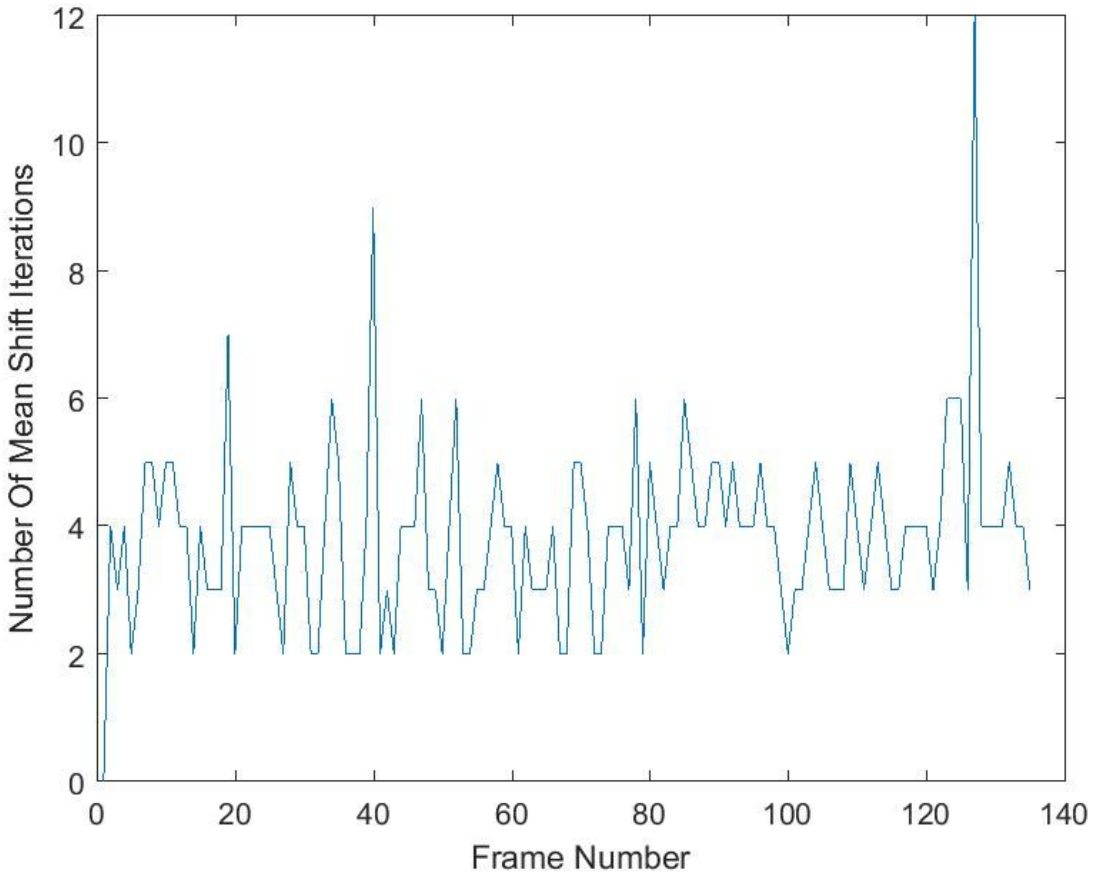
FRAME 200

FRAME 250

FRAME 300

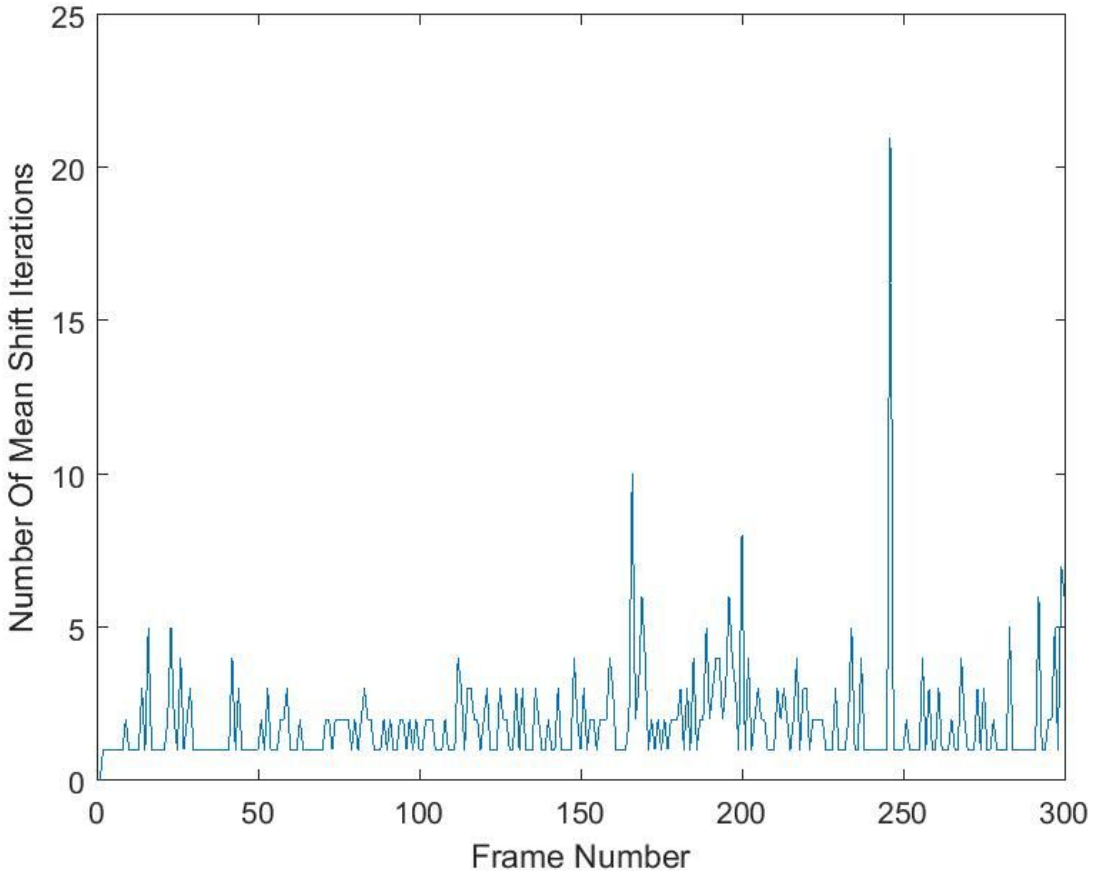
Number of Mean Shift Iterations vs. Frame Index:

COPS CHASING MAN-



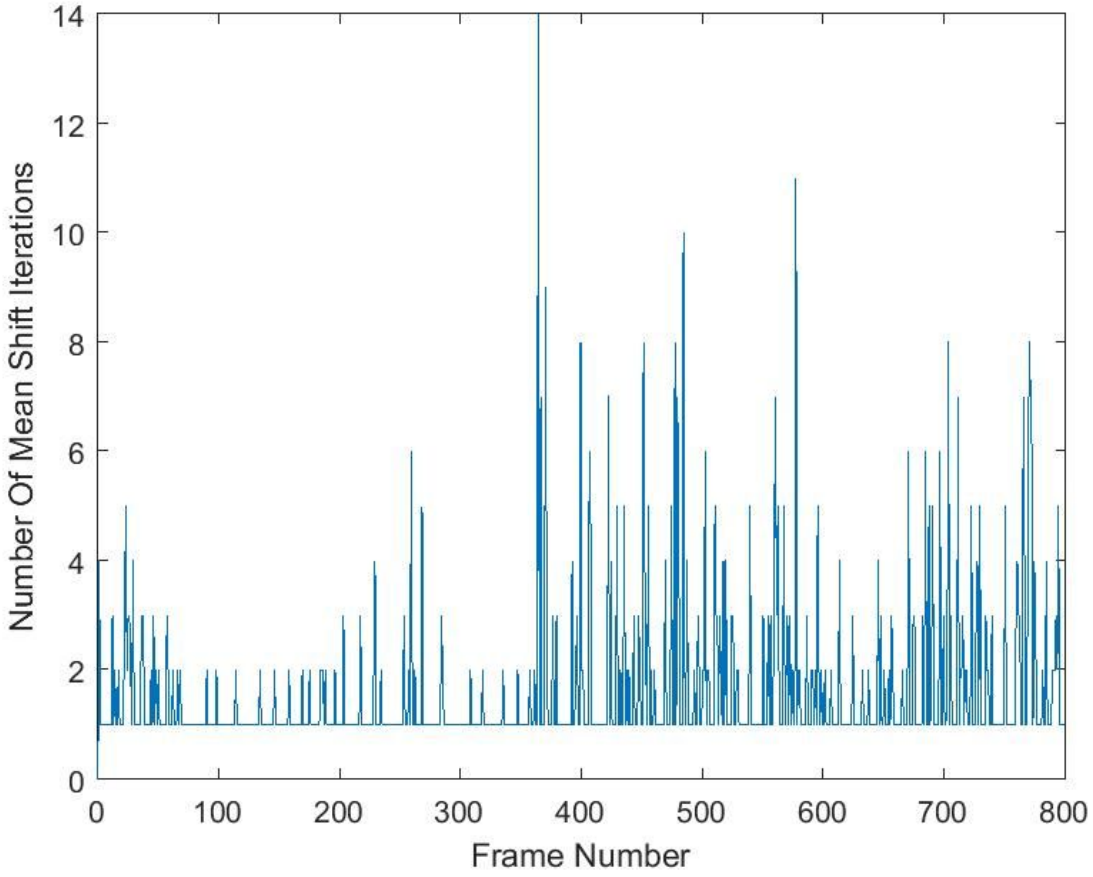
Mean number of iterations = 3.8582

SHOPPING SEQUENCE 2-



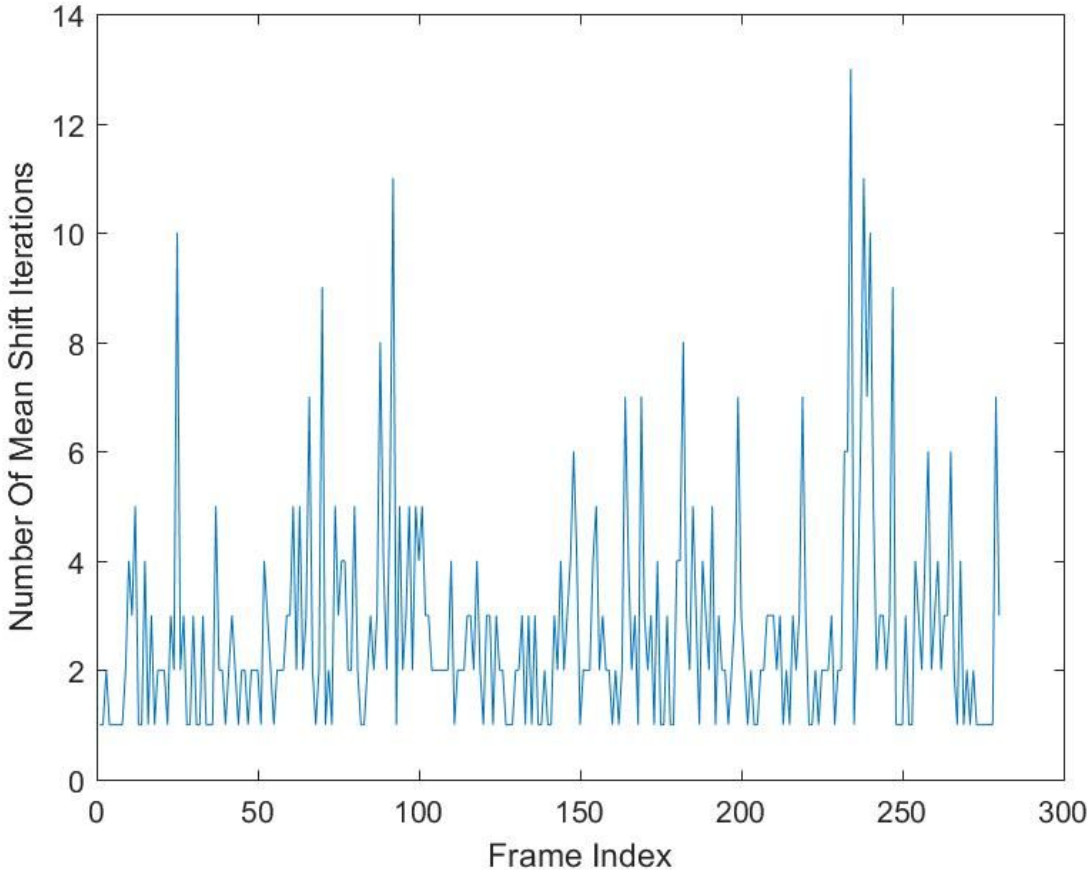
Mean number of iterations = 1.8963

SHOPPING SEQUENCE 1-



Mean number of iterations = 1.5895

FOOTBALL SEQUENCE -



Mean number of iterations = 2.7036

OBSERVATIONS AND CONCLUSIONS:

1. The paper suggests 2 kernels - the Epanechnikov kernel and the Normal kernel. In almost all our experiments, the Epanechnikov kernel had convergence issues. However, the Normal kernel converges on mostly all instances and hence it was our preferred kernel.

2. In some cases, when we chose the window size to be small, there were convergence issues. Similarly, if a large window is drawn around the object, then also the algorithm fails. Hence, it is recommended that a reasonably sized window enclosing the object (to be tracked) be chosen.

3. In cases where the object to be tracked and it's background have similar intensity patterns, the algorithm fails obviously.

4. The algorithm is resistant to partial occlusions as mentioned in the paper.

5. The algorithm also works decently even for poor quality videos, i.e. videos taken with low resolution cameras, noisy as well as blurred videos.

6. For most videos, the average number of mean shift iterations is not very large due to which the algorithm works reasonably fast. In our plots, the average number of mean shift iterations was not more than 4 in any case.

7. The radius of the kernel 'h' as mentioned in the algorithm, needs to carefully tuned for good performance. This parameter is necessary to make the algorithm invariant to scale changes. The paper suggests to use 'h' values within $\pm 10\,\%$ and choose the radius that yields the

maximum decrease in the distance. However, we concluded that it is better to use a margin of $\pm 20-25\%$ especially for fast moving objects.

8. The choice of the bin size for computing the histogram of the target model and the target candidate is also important. For most applications, a bin size of 8 should suffice. However, for some extreme cases in which the intensity of the object being tracked and the background are similar, one might want to reduce the bin size to 4 for better tracking. Unless absolutely necessary do not reduce the bin size, as it significantly increases time as well as memory requirement.

9. The entire mathematical derivation of this algorithm is based on the assumption of elliptical (or circular) windows. We have used rectangular windows in our implementation which work just as well. Note that proper normalization of the x and y coordinates of the location of the pixels is also very essential, else the kernels give very small values and the algorithm doesn't work very well.

DIFFICULTIES:

1. A lot of time was spent on finding appropriate videos on which we can test the algorithm. The object to be tracked should not be very small in comparison to each frame of the video. There should not be abrupt changes in the location of the object being tracked and the intensity of the background should be different from that of the object. Also, the video should have one continuous shot of the object under consideration.

2. In some cases, the tracker follows the object perfectly until a certain frame and after that it just loses track of the object. Beyond that there is no hope of recovery! We had to carefully choose the window size and the radius ‘h’ of the kernel in order to mitigate these effects.

3. In many videos, the object being tracked does not appear immediately when the video starts. In this case, we had to extract the frame in which the object appears for the first time in the video/the size of the object is large enough for our algorithm to work satisfactorily.

4. We tried the Epanechnikov kernel too in our test cases. In most cases, the mean shift iterations do not seem to converge when we use the Epanechnikov kernel. On the other hand, the Normal kernel usually converges quickly enough. However, in some cases even the Normal kernel was unable to converge.

5. As mentioned in the previous point, there were convergence issues in many cases. One may reason that it was due to abrupt changes in the position of the object in which case, the Taylor Series Expansion of the Bhattacharyya coefficient only up to the first term fails. However, the cases we tested on involved no such abrupt changes and we could not figure out the reason as to why this was happening.