Homework 06

Deepak Sharma
CSCI 731 Advance Computer Vision

August 9, 2017

Survey of Tracking Algorithms.

1. **Proposal**
   There are different object tracking algorithms available in OpenCV [8]. I want to conduct a survey study to understand the working and limitation of these algorithms. I will implement a small wrapper code which will facilitate me to invoke these existing tracker algorithm and to apply these algorithm on different test videos where target object will show different types of motions and transformations.

2. **Objectives**

   a. **Literature Survey:** explain working of existing tracking algorithms: BOOSTING, MIL, KCF, TLD, MEDIANFLOW.

   b. **Programming:** Implement a wrapper application for selecting input video, tracking algorithm and taking user inputs for executing tracking.

   c. **Evaluation:** Finding videos/dataset where target object shows: in plan rotation, out of plan rotation, transformation, scaling, partial occlusion and re-appearance.

   d. **Results:** Reporting limitations of above mentioned existing tracking algorithms.

   e. **Future Direction:** In order to stay par with latest development in the filed, reporting the recently proposed state of the art tracking algorithm which address the limitation found in above mentioned algorithms.

3. **Literature Survey:** Object tracking is one the fundamental problem of the computer vision. A plethora of work has been done by a number of researchers in this direction. MEDIANFLOW, BOOSTING, MIL, TLD and KCF are some of the well known trackers which have produced state of art results in their respective years of publication. Now we will discuss each algorithm briefly.

   a. **MEDIANFLOW:** OpenCV median flow algorithm is the implementation of [9]. Median flow algorithm is based on Lucas Kanade feature point tracking method where points associated with the target object are tracked in the sequence of frames. Kalal et al. suggested improvement in tracking quality by removing outlier points. In order

to remove outlier points, each tracked feature point in current frame was backtracked in previous frame. If backward tracked point was mapped far away from the location original associated point(during forward tracking) in the previous frame then such point was identified as outlier.
During tracking location of object is defined by the median location of all points associated with the target object. Scale transformation in each dimension is measured using ratio of distance between each pair of points in consecutive frames.
This algorithm does not address real world tracking problems like appearance changes, disappearance, reappearance of target point.

b. **BOOSTING:** OpenCV boosting algorithm is the implementation of [4]. Grabner and Bischof have address the tracking problem through a adaptive boosting classification algorithm. During tracking, appearance changes of target object was addressed by an online learning algorithm which adjust its learned parameters for incorporating change in appearance. Similarly algorithm also update its knowledge about dynamic background around the target object, thus algorithm also shown additional robustness against background changes.
Initially classifier train itself by the positive(foreground) patch which overlaps the target patch and negative patches extracted from the neighborhood of the foreground patch. Classifier use haar like features or HOG features associated with patches for training. In the next frame classifier predict label for patches extracted from the nearby location of the object in the previous frame. New object location overlap assign to the patch location witch received positive label with highest confidence. The classifier now retrain itself with new foreground and background training patches located nearby the updated location.
Adaptive boosting classifier consist of multiple weak classifier, at each iteration a selector select one of the member classifier for labeling a patch.

c. **MIL:** OpenCV MIL algorithm is based on the [2]. MIL algorithm is an improved version of the above discussed BOOSTING algorithm. In case of misclassification(drifting) during the execution of Boosting algorithm, newly extracted, incorrect labeled foreground and background patches further deteriorate the performance of tracker. In order to resolve this problem, Babenko et al. used multiple instance learning mechanism where instead of assigning foreground and background label to each patch, collection of patches (bag of patches) were assigned labeled. During training, for foreground multiple partially overlapping patches with target object and for background multiple neighboring patches of foreground were extracted. Now bag of foreground and background patches were assigned to positive and negative labels. Such arrangement provide the tracking algorithm robustness against drift. During experiment MIL approach outperformed boosting approach.

d. **TLD:** OpenCV TLD algorithm is based on the [10]. Above discussed BOOSTING and MIL algorithms assume that target object does not move very far from its location in previous frame and they do not address the real world tracking scenarios like object disappearances and reappearances. Kalal et al. suggested a tracking algorithm which not only explore target object in neighborhood but also using sliding window approach explore in all the patches. If both of these branches fail to report the location of

object then it declare disappearance of the target object. In order to compensate the additional computation, the ensemble's member classifiers perform early rejection of patches which reduce the number of candidate patches for final nearest neighbor classifier.

e. KCF Though, by increasing the online training samples sizes of tracking algorithms improves the accuracy but it also hampers the real time performance of previously mentioned algorithms. Henriques et al. have found that by adding more samples of training, the problem become a circulant problem. They provided a FFT based solution which is much faster than previously discussed tracking methods.

4. **Evaluation and Results** Most of the paper have performed experiments on their own datasets consisting few videos. These datasets were not accessible reproducing the results. However algorithms were tested with diving board videos and all of the above tracking algorithm failed to track rotation of the head of the divers still KCF shown best results. In other videos TLD was able to identify the disappearance of target object due to complete occlusion. In longer videos all algorithms were showing drift KCF provided best results.

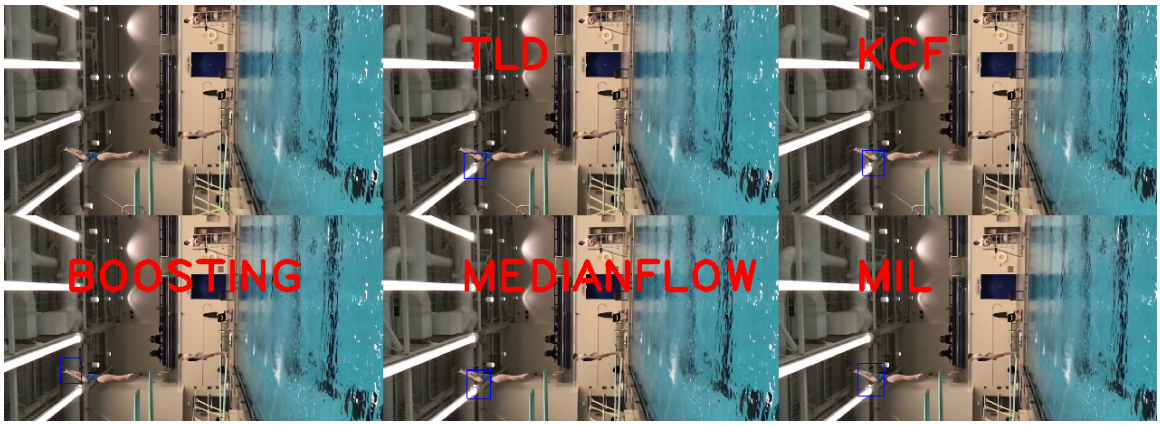

Figure 1: Comparing Tracking Algorithms

5. **Source code credit**
   1. Tracking wrapper [11, 8]
   2. Image operations (Intersection, Union, image stitching etc.) [1, 7].
   3. Codding style and guidelines [3].
   4. Assertion of a file path [12].

6. New **state of art** tracking algorithms are based on DeepLearning like GOTURN[5], but trackers are offline trackers.

References

[1] Gary Bradski Adrian Kaehler. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library 1st*. O'Reilly Media, Inc., 2 edition, 2016. ISBN 1491937998 978149193799.

[2] Boris Babenko, Ming hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning, 2009.

[3] Errin Fulp. C++ coding guidelines, 2006. URL `http://csweb.cs.wfu.edu/~fulp/CSC112/codeStyle.html`.

[4] Helmut Grabner and Horst Bischof. On-line boosting and vision. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, CVPR '06, pages 260–267, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2597-0. doi: 10.1109/CVPR.2006.215. URL `http://dx.doi.org/10.1109/CVPR.2006.215`.

[5] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference Computer Vision (ECCV)*, 2016.

[6] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part IV*, ECCV'12, pages 702–715, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33764-2. doi: 10.1007/978-3-642-33765-9_50. URL `http://dx.doi.org/10.1007/978-3-642-33765-9_50`.

[7] *The OpenCV Reference Manual*. Itseez, 3.2 edition.

[8] *The OpenCV Reference Manual*. Itseez, 3.2.0.0 edition, December 2016. URL `http://docs.opencv.org/trunk/d2/d0a/tutorial_introduction_to_tracker.html`.

[9] Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-backward error: Automatic detection of tracking failures. In *2010 20th International Conference on Pattern Recognition*, pages 2756–2759, Aug 2010. doi: 10.1109/ICPR.2010.675.

[10] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1409–1422, July 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.239. URL `http://dx.doi.org/10.1109/TPAMI.2011.239`.

[11] Satya Malick. Object tracking using opencv, 2017. URL `https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/`.

[12] Pherric Oxide Vincent, Pevik. Fastest way to check if a file exist using standard c++/c++11/c?, 2012.