Long Term Tracking and Detection of Failure

Sravani Teeparthi
Department of Electrical and Computer Engineering.
University of New Mexico
Albuquerque, New Mexico
sssravani4@gmail.com

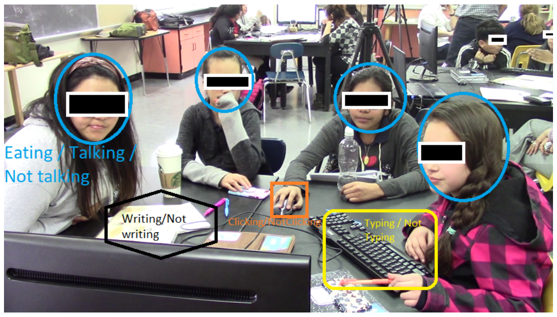
***Abstract*—Human action recognition, as one of the most important topics in computer vision, has been extensively researched during the last decades due to its potential diverse applications where group activities is a very challenging task. Based on data set from AOLME project, the goal of the project is to track people for longer time and detect the failure of tracking.**

I. Introduction

AOLME is an out-of-school learning initiative that promotes middle school students awareness in engineering through the processing of digital images and videos. As part of this program, videos are captured in each session. These are later transcribed and analyzed by learning experts. Having to go through this huge database of videos to capture moments of interest is labor intensive and time consuming. By converting this into video activity recognition problem we would like to facilitate learning experts with land marks and statistics of moments of interest.

II. Motivation

Motivation behind this project is mainly Human Activity recognition.There are multiple activities with multiple people in AOLME videos. Activities be like typing, writing, talking etc. Main focus of doing this project is to classify people as talking and not talking, so for this tracking of head is very important. So it is same as Activity Recognition problem,by converting this into video activity recognition problem we would like to facilitate learning experts with land marks and statistics of moments of interest.



- Activity Localization
- Extracting Spatial and Temporal features
- Classfication

A. Why is tracking important?

- State of the art activity detection algorithms are generally developed on
  - Cropped and/or Trimmed Videos and average play back is less than 12. 8 sec Table II-A
- Moving ROI: Position of object of interest changes with time
  - Spatially fixing the roi will not capture required object after some time
- Multiple activities: AOLME videos (our dataset) has multiple activities occurring simultaneously.

III. Expected contribution

Expected contributions are listed below,

- Long term tracking
- Comparision of available tracking algorithms
- Classifier to detect failure

IV. Dataset

As part of AOLME data set we collected more than 1000 hours of videos having group interactions. These are recorded across multiple groups having different students over time. This provides a diverse dataset for activity recognition. In current experiment we used two videos of one group. These have resolution of 858x480 at 30 FPS (frames per second).

V. Background

As background we have considered all the trackers supported in OpenCV 4.1.0. These are listed below,

1) Boosting [1] - 2006
2) MIL [2] - 2009
3) MOSSE [3] - 2010
4) Median Flow [4] - 2010
5) TLD [5] - 2012
6) KCF [6] - 2015
7) GOTURN [7] - 2016
8) CSRT [8] - 2017

As part of background we have tested all these trackers for speed and long term performance. Table I summarizes speed while Figure 6 provides their performance.

Dataset	Year	No of categories	Mean length of clip	Mean resolution	Mean frame rate
UCF101	2012	101	7sec	320*240	25
HMDB51	2011	51	2 to 5 sec	height=240	30
ActivityNet	2015	203	...	1280*720	30
Charades	2016	157	12.8	320*240	25
HACS	2017	200	2sec	112*114	25
Kinetics600	2018	600	10sec	variable	variable

A. *Speed test*

We used **Xena cluster**, managed by CARC to perform these tests. Summary of system and video properties are provided below,

System

CPU: Intel Xeon CPU E5-2640, 16 Cores per node at 2.6 GHz.
RAM: 64 GB
GPU: Nvidia Tesla K40M

Video

Resolution: 858×480
FPS: 30
Playback time: 23.45 Minutes

OS

Debian stable singularity image
Compiler: g++ 6.3.0

TABLE I
SPEED OF TRACKERS SUPPORTED BY OPENCV. VIDEO UNDER CONSIDERATION IS 858×480 @ 30 FPS. GREEN INDICATES REAL TIME, WHILE RED INDICATES SLOWER THAN REAL TIME

Method	**Time**	**Speed up**
Boosting	40.4 min	$\times 0.5$
MIL	63 min	$\times 0.36$
MOSSE	1.4 min	$\times 16$
Median Flow	3.15 min	$\times 7$
TLD	32.3 min	$\times 0.71$
KCF	4.4 min	$\times 5$
GOTURN	$>$ 1.5 hours	$< \times 0.2$
CSRT	29.9 min	$\times 0.8$

B. *Performance Evaluation of available methods*

Here we tested all these available algorithms in Opencv on the AOLME dataset and plotted their performance evaluation for every second in 23.45min. The metric used for performance evaluation is Intersection over union $P\_suc = r\_t \cap r\_g / r\_t \cup r\_g$; r_t is the bounding rectangle returned by tracker and r_g is the bounding rectangle by ground truth, Fig 6. Ideal value is 1.

VI. METHODOLOGY

Figure 1 provides an overview of method used in this paper. We started experimenting with various concepts such as,

1) Normalized Cross Correlation
2) Normalized Cross Correlation with template update for every frame
3) Normalized Cross Correlation with moving window over search region
4) Normalized Cross Correlation with moving window and template update

before we developed our own method. By these experiments we understood that normalized cross correlation with failure detection and update works in our case.

We start with video and bounding box on first frame. In next frame we use the object inside the bounding box as template and perform template matching using normalized cross correlation (nxcorr). Which ever position gives highest value within a search window is taken as best matching case. We put this new template through a classifier that detects if this new bounding box was a success (s) or a failure (f). If it is classified as failure we further process try to fix the bounding box using previous ‘n’ successful tracking.

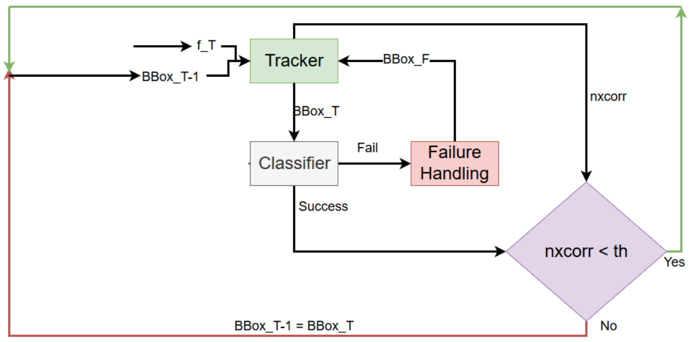


Fig. 1. Performance of Trackers

VII. RESULTS

The method which gave best results in this case is Normalized Cross Correlation with restricted search. Figure 2 compares IoU scores between not updating template and updating template, while figure 3 shows result after failure handling.

From this figure, you can see Normalized Cross Correlation is performing better.

A. *Performance of Normalised Cross Correlation Vs other trackers*

From figure 4, We can see that Normalized Cross Correlation is performing very well. Here for comparision Ground Truth is done manually for 23min video for every second

For training a classifier we used bounding boxes from ground truth and the output of normalized cross correlation and then compared them with each other and classified them as **failure** and **success** manually for every second.
We trained it on one video of 23.45min length which has 161

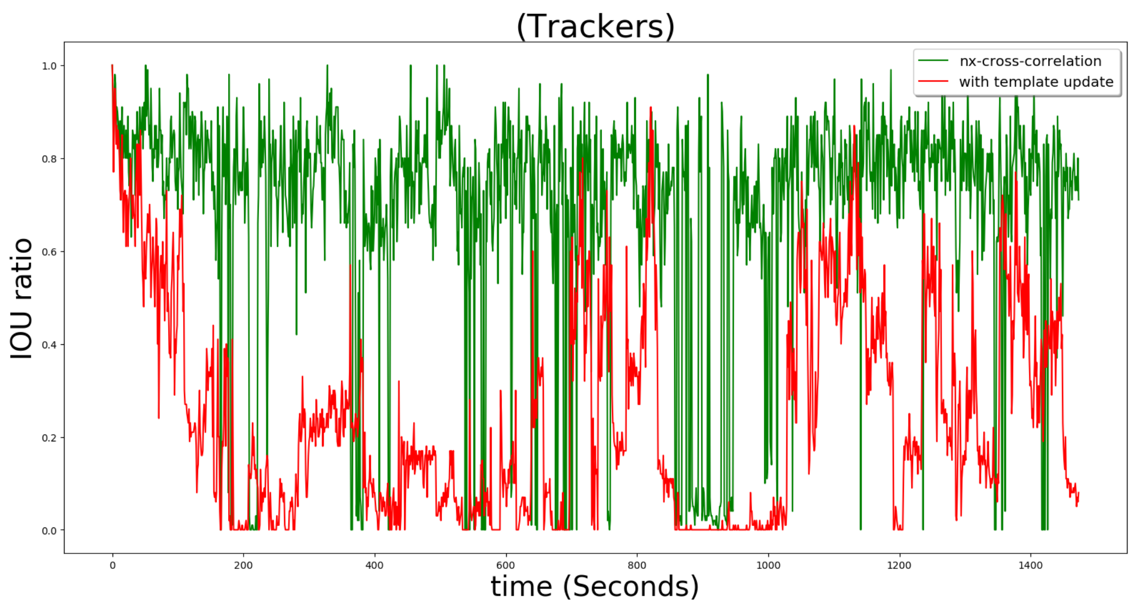


Fig. 2. Performance of Normalized Cross Correlation and with template update

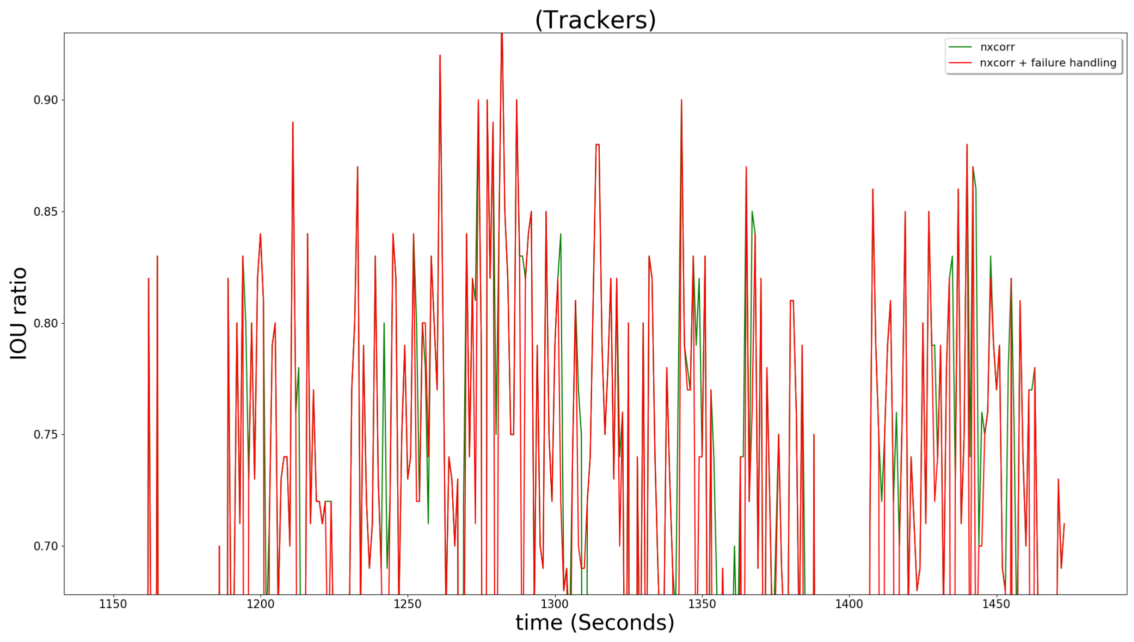


Fig. 3. Handling failure by looking into past (180 succesful frames). Here we do see that we did not improve much compared to the base method.

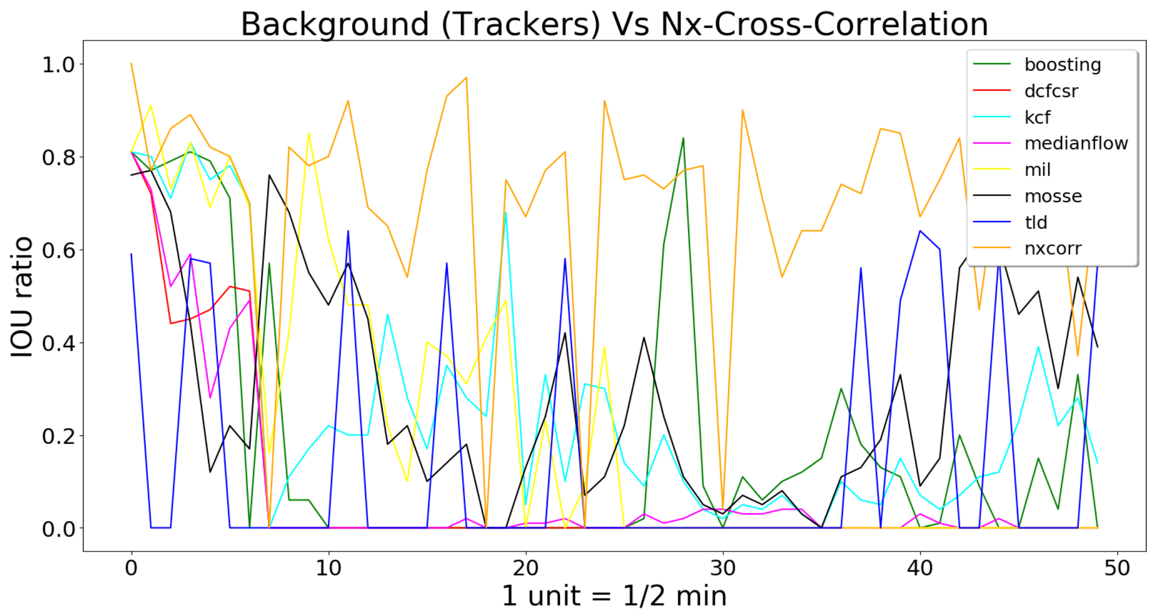


Fig. 4. Performance of Normalized Cross Correlation Vs all other trackers

classes as failure and 1313 classes as success. The regions are cropped using the bounding boxes from ground truth and the method from the video and then they are converted to gray scale.

Then histograms is taken with bins=20 for both the images and difference of histograms is saved as numpy arrays for both success and failure. We used different videos for training and testing. The testing dataset has 1474 bounding boxes from other video. Figure 5 shows ROC and AUC for Random forest classifier.

TABLE II
RANDOM FOREST WITH 20 FEATURES EXTRACTED FROM HISTOGRAM OF GRAY IMAGE (Y). NUMBER OF ESTIMATORS = 30, DEPTH = 13. WE RECEIVED AN ACCURACY OF 80% ON TESTING.

Manual	**Proposed Method**		
	failed	success	Total
failed	**55**	233	288
success	27	**1159**	1186
Total	82	1392	1474

VIII. CONCLUSION / FUTURE WORK

1) Create better failure detection and recovery algorithm.
2) Running on bigger dataset.
3) Detect occlusions and object missing from frame.
4) Supporting moving window and hence detecting when and object leaves field of view.

REFERENCES

[1] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting." in *Bmvc*, vol. 1, no. 5, 2006, p. 6.

[2] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 983–990.

[3] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2544–2550.

[4] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *2010 20th International Conference on Pattern Recognition*. IEEE, 2010, pp. 2756–2759.

[5] ——, "Tracking-learning-detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.

[6] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[7] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 749–765.

[8] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6309–6318.

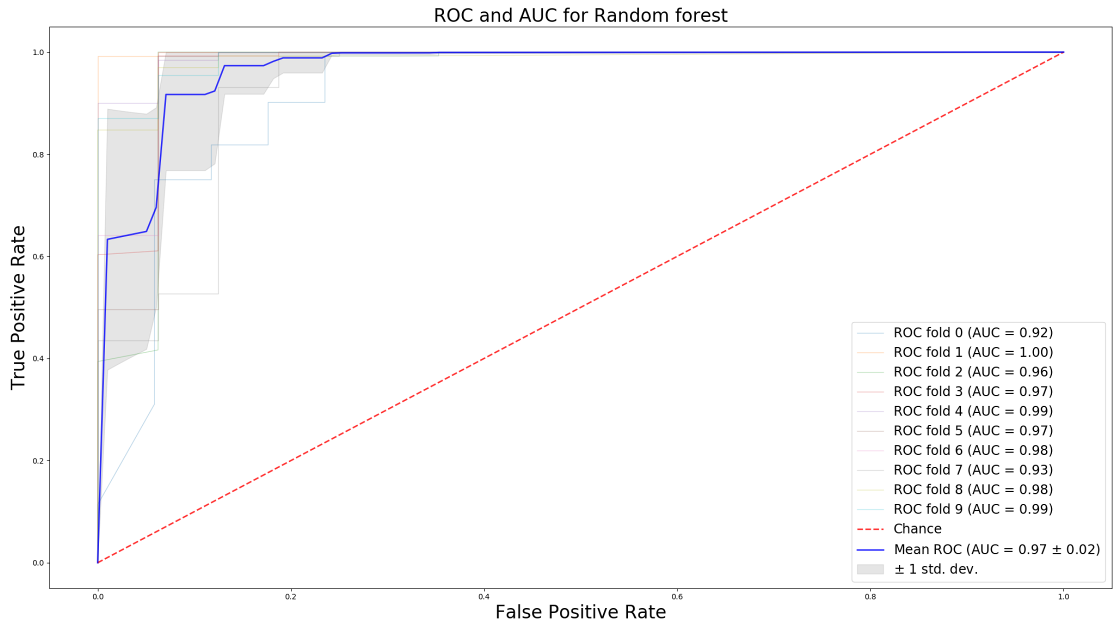


Fig. 5. plot of ROC and AUC for Random Forest

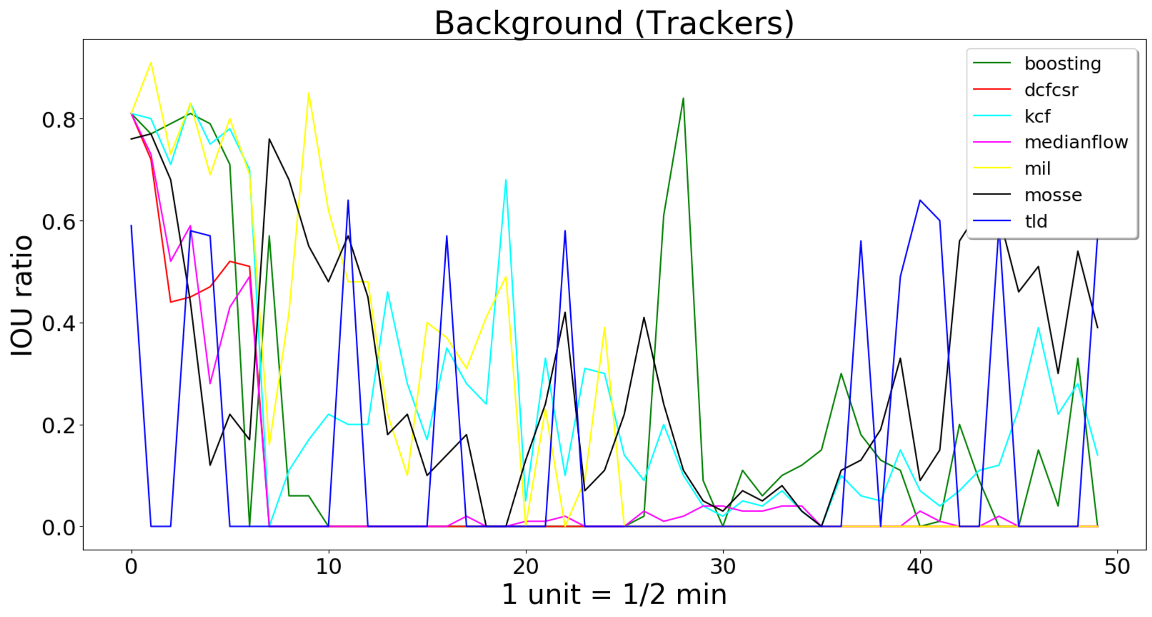


Fig. 6. IoU performance of trackers supported by OpenCV