Comparison of tracking algorithms implemented in OpenCV

Aleksandra Knapińska
Wrocław University of Technology
Faculty of Electronics
Wrocław, Poland
226057@student.pwr.edu.pl

Mateusz Janiszewski
Wrocław University of Technology
Faculty of Electronics
Wrocław, Poland
226066@student.pwr.edu.pl

***Abstract*—Tracking algorithms can be used for various applications like traffic control, face recognition, image matching etc. There are several trackers implemented in a popular computer vision library OpenCV. In this paper they are compared in tracking objects of different shapes. TLD tracker appeared to be the most accurate.**

***Index Terms*—OpenCV, tracking algorithms**

This paper is based on the "Comparison of tracking algorithms implemented in OpenCV" paper by Janku, Koplik, Dulik and Szabo [1].

I. Introduction

Locating an object in successive frames of a video is called tracking. Object tracking is an important issue in computer vision. The tracking algorithms are widely used in many areas e.g traffic control. There have been created many algorithms which deal with tracking object. However, it is not possible to choose the universal tracking algorithm - the most effective one. The reason is the fact that each algorithm has to deal with a big number of parameters like illumination, background, shape of objects etc. What is more, the performance of the tracking algorithm is connected to the algorithm's implementation. The key point is choosing the right programming language. In order to omit the mentioned problem, we decided to use tracking algorithms from OpenCV library.

II. Algorithms description

The OpenCV library contains the following tracking algorithms:

A. BOOSTING

The tracker is built on AdaBoost classifier. The algorithm must be trained at runtime - providing positive and negative examples of a tracked object. The area to be tracked - (bounding box - marked by user) is treated as a positive example and images outside of the box is a background. In the new frame, the classifier is executed on every neighbourhood pixel of the previous object's location and a score of the classifier is set. The updated position of the object is designated based on the highest score of the classifier which is set just before. Every time the object changes position, the algorithm has a new positive example of the classifier. Thus, in time algorithm has additional data which helps in tracking. The algorithm is quite old and the quality of tracking is mediocre.

B. MIL

The tracker works similar to BOOSTING. The main difference is that instead of collecting the current position of a tracked object as a positive example, the algorithm scans close neighbourhood around the current location to create some potentials positive examples. The collections of positive and negative examples can be described as 'bags'. As mentioned before, the collection of positive examples contain only one positive example. Other are false examples. Thanks to the approach of positive 'bags', the position of the tracked object might not be accurate. There is a huge chance that 'bag' contains at least one image on which object is centred.

C. KCF

The name of the tracker is Kernelized Correlation Filters. The idea behind this tracker is based on BOOSTING and MIL. The tracker uses the fact that the multiple positive examples (used in MIL) have overlapping regions. Based on that, the tracker uses mathematical properties used in formulas to make the tracking faster and more accurate at the same time. The drawback is that implementation in OpenCV 3.0 does not contain recover from full occlusion.

D. TLD

The steps of the algorithm are the following: tracking, learning and detection. The tracker follows the object from frame to frame. The tracker collects all positions of objects and corrects the tracker if it is necessary. In the learning step, the algorithm estimates the detector's errors and updates it to prevents these errors in future. The tracker tends to jump around the target e.g if a specific pedestrian is tracked among other pedestrians, the tracker can change focus on another pedestrian.

E. MEDIANFLOW

The tracking algorithm tries to predict the tracked object position in both forward and backward directions. Then, the discrepancies between these two trajectories are measured. The minimization of Forward-Backward error enables finding tracking failures and track the object properly.

F. MOSSE

Minimum Output Sum of Squared Error uses adaptive correlation, which produces stable correlation filters when initialized using a single frame. The tracker is also able to pause when the object disappears and resume where it left off when the object reappears. It also operates at a higher fps, which makes it possible to track fast moving objects.

G. CSRT

The acronym stands for Discriminative Correlation Filter with Channel and Spatial Reliability. This tracker uses the spatial reliability map for adjusting the filter support to the part of the selected region from the frame for tracking. It operates at a comparatively lower fps but gives higher accuracy for object tracking.

The trackers are described in more detail by Gazar in [2].

III. Measurement methods

To measure the success rate for each tracker, two time-lapse videos were created, both consisting of ten frames. Each video has one object which changes its position in every frame. The background is white. The reason to create two videos instead of one is to check how the trackers perform with objects of different shapes. The object to be tracked in the first video is a pen, while in the second video it is a Barbie doll wearing a crown and a mermaid tail. The objects in the videos do not change their orientation (do not rotate), they just change the position.

To measure the performance of the trackers, a simple script was created. It works as follows: first, a video is imported. It is frozen on the first frame. Then the user has to draw a rectangle around the object to be tracked. After that, the video starts playing and the rectangle should start moving with the tracked object. The success is measured as a number of frames in which the object was inside the rectangle. Each video was tested ten times for each tracker.

IV. Results

The figures 1 and figure 2 present the success of each algorithm. The 'Success value' of each algorithm is an average value in 10 measurements. Each measurement can return value in range 0 - 1, where 0 means that the tracking algorithm does not track the object in any frames, 1 means that tracking algorithm tracks object in all frames.

The figure 1 shows that the most accurate algorithm is TLD. The second place is occupied by the CSRT algorithm. The weakest accuracy is assigned to MOSSE algorithm.

The figure presents that Boosing, TLD and CSRT return the most accurate results. The poorest accuracy is returned by KCF algorithm.

V. Conclusions

The TLD tracking algorithm turned out to return the most accurate results. It correctly tracked the object in both videos with the best success rate.

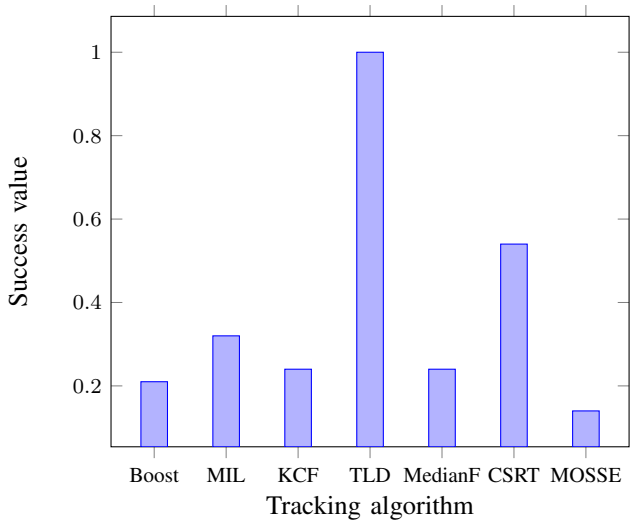

Fig. 1. Success value dependence of tracking algorithm for "Pen video"

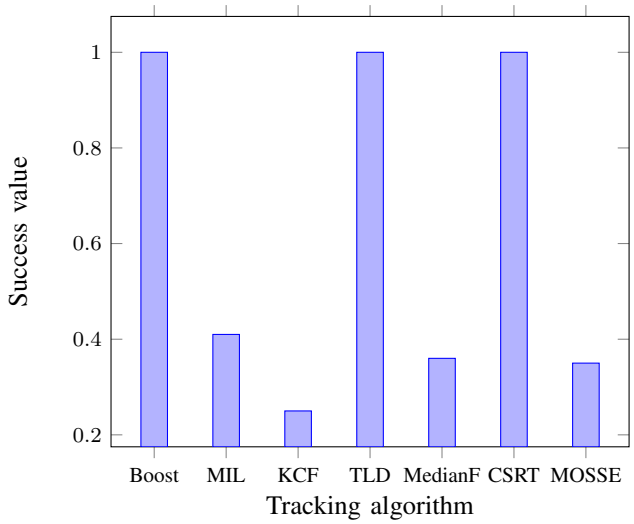

Fig. 2. Success value dependence of tracking algorithm for "Barbie video"

The worst tracking algorithms appeared to be KCF and MOSSE. They both had less than 0.4 success rate in both videos.

Another fact is that "Barbie doll" was properly tracked by more tracking algorithms than "Pen". It can be caused by the fact that the "Barbie doll" has an irregular shape and has more colours than "Pen".

References

[1] P. Janku, K. Koplik, T. Dulik, and I. Szabo, "Comparison of tracking algorithms implemented in opencv," in *MATEC Web of Conferences*, vol. 76, p. 04031, EDP Sciences, 2016.

[2] E. Gazar, "Object tracking with opencv," accessed June 2020. https://ehsangazar.com/object-tracking-with-opencv-fd18ccdd7369.