Tracking API

- Author: Ilya Elizarov
- Link: #18481
- Status: **WIP**
- Platforms: **All**
- Complexity: N/A

Introduction and Rationale

The main goal of this proposal is the renewal of the tracking module which has existed in stagnation in opencv-contrib for a few years and moving it to the OpenCV main repository.

For now, we have 8 trackers in the "opencv_contrib" repository (7 classical CV, 1 DL-based):

1. MIL [17]
2. Boosting [18]
3. MedianFlow [19]
4. CSRT [20]
5. KCF [21]
6. TLD [22]
7. MOSSE [23]
8. GOTURN [24]

Also, 2 modern DL-based trackers are implemented as Python 3 samples in "opencv/samples/dnn":

1. DaSiamRPN [26]
2. SiamRPN++ [27]

In the future, it is planned to add C++ implementations of these trackers in the tracking API. The main OpenCV repository contains several basic blocks that can be used in custom tracker implementations prepared by users. They include:

1. Optical flow algorithms (Sparse or Dense)
2. MeanShift
3. CamShift

Main steps for the renewal:

1. Improving tracking API for more convenient work with classic and DL-based trackers at the same time.
2. Moving tracking module from opencv-contrib to the main repository (most of the trackers) under the same name or extending the functionality of the existing "video" module (old "tracking" module can be preserved for "opencv-contrib" experimental algorithms).

Roadmap after renewal:

1. Add C++ implementations of the DaSiamRPN and SiamRPN++.
2. Fix issues with multi-object tracking.
3. Create a benchmark for multi-object tracking (preferably MOT-based [28] for more convenient comparison of the results).
4. For the last half-year, I worked on "opencv-contrib" trackers: tried to check accordance with the original papers, writing benchmarks on Python (based on LaSOT [11] and TrackingNet [12] metrics). The main problem was the absence of updates in the module.

The rationale for moving the module to the main repository. In September 2020, we have 24 opened / 21 closed issues and 5 opened / 39 closed PR's related to trackers (time range 5 years ago - present days) [8][9]. And some of them are really old (opened years ago). It shows 2 important things:

1. The module still in demand by the community
2. The module has low priority for the OpenCV development team

As a result of that, we have no new trackers and we have issue reports with no answers. But tracking is a very popular computer vision task now, and we can see different ways to solve it without tracking module [1][2][3]. If we will try to find some sort of "from the shelf" solution, we can see that OpenCV is a popular solution for tracking, despite all its problems [4][5]. Important numbers here - number of the views, it is counted by thousands. Also, worth mentioned fact - the community is still creating guides and tutorials for our old trackers [6][7][8].

Choosing trackers for moving to the main repository. We should move GOTURN, MIL, and KCF trackers. For now, GOTURN is the only one DL-based tracker in the module. MIL and KCF trackers are still competitive compared with modern trackers as shown in TrackingNet, LaSOT papers, due to VOT 2017-2020 results [13], and research papers [16]. Their main pros for us - they are working on CPU, while other top-tier trackers working on GPU.

Proposed solution

For now, the benchmark, which uses modern generally accepted LaSOT and TrackingNet metrics are completed (results are in the description of the PR) [25]. It measures precision, normalized precision, and intersection over union for all 8 trackers in opencv-contrib and for the DaSiamRPN sample. Also, I and Dmitry Kurtaev(@dkurt) fixed an old memory problem for GOTURN tracker.

Proposed steps:

1. Move the tracking module to the main repository
2. Add some changes in the API (reinitialization of the trackers, work with classic and DL-based trackers in the same API)

Reinitialization problem: then I tried to reinitialize the tracker after object loss, I saw that we can not put another bounding box in the already existing

tracker. A similar problem we can see in one of the old PR's [14]. I suggest changing the "init" method for trackers.

Classic vs DL-based problem: for now, we need both (at least until we have more than 1 DL-based tracker in the module). KCF and MIL trackers are actively used by the community, and showing satisfactory results compared with modern trackers. They should be a good base for a "new" module.

Impact on existing code, compatibility

Tracking API is independent, and changes in it should not affect the rest of the library. Changes in the initializing method should not be redundant - we need only add some functionality for reinitialization. Changes relative to DL-based trackers should facilitate the process of the loading models of the trackers.

Compatibility with previous versions of the trackers will be lost:

1. We will remove some of them
2. The rest of them we will move into the another repository
3. We will change the initialization method
4. We will add functionality for DL-based trackers

But more important is that:

1. We try to create new, more convenient API for all kinds of trackers
2. We want to attract new developers in our community, and it is gonna be easier with these changes
3. We will create a platform for adding new modern trackers
4. We will save the most valuable trackers

Possible alternatives

As an alternative, we can move only the GOTURN tracker as the DNN High-Level API algorithm.

Pros: 1. Popular in community and for researchers 2. Orientation to modern trackers 3. No need to change the API for the convenient use of classic and DL-based trackers at the same time.

Cons: 1. Risk of losing the rest part of the community - many of them using classic trackers 2. "Cut off" classic tracker, which can be better in real-life cases, production 3. DL-based trackers still show instability compared to classic trackers. For example, GOTURN is not so popular and a robust tracker - its LaSOT results are worse than the results of the classic trackers [15].

References

1. https://medium.com/milooproject/object-tracking-using-opencv-python-windows-616fb23da720

2. https://www.youtube.com/watch?v=bSeFrPrqZ2A
3. https://www.youtube.com/watch?v=19vaot75JCY
4. https://www.youtube.com/watch?v=61QjSz-oLr8&feature=emb_logo
5. https://www.youtube.com/watch?v=1FJWXOO1SRI
6. https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/
7. https://www.coursera.org/projects/computer-vision-object-tracking-opencv-python
8. https://www.pyimagesearch.com/2018/07/30/opencv-object-tracking/
9. https://github.com/opencv/opencv_contrib/pulls
10. https://github.com/opencv/opencv_contrib/issues
11. https://arxiv.org/abs/1803.10794
12. https://arxiv.org/abs/2009.03465
13. https://prints.vicos.si/publications/groups/vot
14. https://github.com/opencv/opencv_contrib/issues/1465
15. https://github.com/opencv/opencv_contrib/pull/2516
16. https://www.researchgate.net/publication/317803149_Evaluation_of_Visual_Tracking_Algorithms_for_
17. https://ieeexplore.ieee.org/document/5674053
18. https://www.researchgate.net/publication/221259753_Real-Time_Tracking_via_Online_Boosting
19. https://ieeexplore.ieee.org/document/5596017
20. https://arxiv.org/abs/1611.08461
21. https://ieeexplore.ieee.org/document/6909539
22. https://ieeexplore.ieee.org/document/6104061
23. https://ieeexplore.ieee.org/document/5539960
24. https://arxiv.org/abs/1604.01802
25. https://github.com/opencv/opencv_contrib/pull/2516
26. https://arxiv.org/abs/1808.06048
27. https://arxiv.org/abs/1812.11703
28. https://arxiv.org/abs/2003.09003