A Benchmark on Modern Tracking Frameworks

Minci Zhou [{*minci@ualberta.ca*}](mailto:%7bminci@ualberta.ca%7d)

Guideline:

1. Abstract. Talk about importance of tracking; talk about how different platforms and different implementation impact tracking performance; talk about a consistent tracker benchmarking platform which can be beneficial for evaluating tracker performance; talk about OpenCV as a popular framework for trackers, their trackers can be used out-of-box and constantly updating; so we benchmarked OpenCV trackers, present results, discuss further implementation of a tracker benchmarking framework which encapsulate individual tracker implementations into OpenCV-like tracker frameworks.
2. Introduction

Object tracking has been an important field of study in computer vision. Many tracking algorithms have been developed but implemented on different frameworks and languages, which hinders other researchers from comparing one algorithm to another, and increases difficulties of evaluating them in a relatively fair environment. Furthermore, the performance of the same algorithm may vary under the same programming language due to rapid progress in hardware performance and compiler optimization in the recent years. [Peter Janku]

OpenCV is a popular open-source computer vision framework capable of performing real time computer vision tasks. It is implemented in C++ for maximum performance boost, although a Python wrapper is also available with little or no performance loss comparing to naïve OpenCV implementation. [Does performance differ between Python or C++ coding of OpenCV?] In this paper, implementations of object tracking algorithms from OpenCV will be compared based on a popular tracking benchmark dataset [Yi Wu], and the results of comparison and possible integration of other object tracking algorithms/frameworks will also be discussed.

1. Similar work

Compare to Peter Janku’s paper, this paper includes benchmarks on newly added algorithms in OpenCV 4 such as MOSSE, CSRT, KCF while removed other poorly performed and patented algorithms such as ORB, SIFT, and SURF. This paper also uses average IOU, including the IOU under a failed frame.

1. Implementation (OpenCV)
2. Discuss the architecture of a sample tracker

In OpenCV 4, new trackers, such as CSRT, KCF, MOSSE, and GOTURN were added. OpenCV integrated them into an easy-to-use, pipeline-like framework, where many of them can be used out-of-box while learning-based algorithm such as GOTURN and Boosting requires pretrained model. For consistency, in this paper GOTURN is not used as it relies on external pretrained model and current implementation of GOTURN has memory leak problem which will be fixed in a short period of time.

1. Implementation (Benchmark)
2. Benchmark initializes a tracker, a dataset, feeds data in loop, compute and record iou and fps;
3. Probably include a pseudo code?
4. Evaluation method
5. Results
6. Conclusion
7. Final thoughts (benchmarking registration-based tracker)
8. Reference

Comparison of tracking algorithms implemented in OpenCV

Object Tracking Benchmark

For results in histogram, remember to plot error bars; x/y axes and labels; use different colours;

Benchmarking on accuracy: IOU;

Benchmarking on speed: fps;