ISSN 1870-4069

Updating and Evaluating the Struck Tracker: A Comparative Study on the ALOV300++ Benchmark

Víctor Alejandro Méndez-López, Carlos Soubervielle-Montalvo, Cesar Puente-Montejano, Rafael Peña-Gallardo, Emilio Jorge González-Galván

Autonomous University of San Luis Potosi, Faculty of Engineering, San Luis Potosi, Mexico
a242881@alumnos.uaslp.com,{carlos.soubervielle,cesar.puente,rafael.pena,egonzale}@uaslp.mx

Abstract. Visual Object Tracking (VOT) is focused on tracking moving objects across sequences of video frames. A VOT system consists of an integrated framework that combines models, algorithms, and hardware to ensure robust tracking performance despite changes in environmental conditions, lighting, object appearance, and other factors. This work presents the modification and evaluation of the Struck tracker to ensure compatibility with the OpenCV 4.9 library, as the original implementation was developed using OpenCV 2.4. This update is reported as a first step toward the implementation of the Struck algorithm on a SoC-FPGA platform as part of an ongoing Ph.D. research project. A series of performance experiments were conducted using all categories of the ALOV300++ benchmark dataset to compare the updated Struck tracker against three baseline OpenCV-based trackers: CSRT, MIL, and KCF. Experimental results demonstrate that the updated Struck tracker achieved the highest F1-score (0.720 ± 0.122) and precision (0.647 ± 0.139) among the evaluated algorithms, with a processing speed of 77.1 FPS. These findings highlight the trade-offs between tracking accuracy and computational efficiency, and demonstrate the feasibility of updating legacy tracking code for use with modern computer vision frameworks and datasets.

Keywords: Visual Object Tracking, Struck Tracker, ALOV300++.

1 Introduction

Visual object tracking (VOT) is defined by the analysis of video sequences for the purpose of establishing the location of the target over a sequence of frames starting from the bounding box given in the first frame [1, 2]. It is an active research field with relevant applications in different domains, including surveillance, robotics, autonomous vehicles, and augmented reality [3]. At the same time, it is a challenging task due to the computational resources needed to keep a real-time response while dealing with a series of diverse factors like complex object shapes, irregular movements, scene illumination changes, and object occlusion, among others.

A variety of paradigms, design methodologies and hardware platforms have been proposed in the literature to address the design complexity of efficient VOT systems

[4, 5]. Among these, the statistical reference model introduced by Smeulders et al. [1] stands out for emphasizing the importance of incorporating predictive models—particularly for motion estimation—rather than relying solely on appearance-based representations. This perspective has driven the development of adaptive tracking mechanisms, which continuously update model parameters and internal representations to maintain robust tracking performance under dynamic conditions [6].

Building on this need for adaptability, two prominent paradigms have emerged in the literature: (a) tracking-by-detection, in which tracking is formulated as a repeated detection problem: in each frame, a classifier is trained or updated to detect the target object in the subsequent frame, typically within a predefined search region; and (b) correlation filter-based tracking, in which tracking is achieved by learning a discriminative filter to locate the target object by measuring similarity (correlation) between a target template and candidate regions in subsequent frames[7]. These paradigms reflect different strategies for incorporating temporal changes into the tracking model.

Within this framework, a subset of VOT algorithms known as online learning has proven especially well-suited for large-scale and non-stationary tracking tasks, as it processes data sequentially and updates models in real time. However, despite its adaptability and predictive capabilities, the kernelized variant of this approach incurs significant computational costs, primarily due to the overhead associated with maintaining and updating kernel-based structures-posing notable challenges in resource-constrained environments, such as embedded systems or FPGA-based platforms[8, 9].

In this context, a subset of representative VOT algorithms has been selected to support the objectives of this study. These include Struck, CSRT, MIL, and KCF, each embodying distinct approaches to model updating, feature representation, and computational complexity. The Struck algorithm combines structured support vector machines with kernel-based methods for tracking. It formulates tracking as a structured output prediction problem and learns a discriminative tracker by leveraging appearance and motion cues[10]. The CSRT algorithm integrates channel and spatial reliability to enhance robustness against occlusions and background clutter. It employs a kernelized correlation filter approach in a multi-channel environment to achieve accurate object tracking [11]. The MIL algorithm models tracking as a binary classification problem using multiple instance learning. It selects the most probable positive instance within a set of candidate instances to track the target object robustly under varying conditions [12]. The KCF algorithm uses kernel methods to learn correlation filters in the Fourier domain. It efficiently computes the correlation between the target object and candidate patches in each frame, enabling real-time tracking with high accuracy [13]. All selected trackers were evaluated using the One-Pass Evaluation (OPE) protocol, with average Precision, F1-score, and Frames Per Second (FPS) computed for each video sequence, as defined in the ALOV300++ dataset by Smeulders et al.[1].

This work presents a series of comparative experiments involving four OpenCV-based trackers—Struck, CSRT, MIL, and KCF—aimed at assessing and discussing their relative tracking performance under diverse visual conditions. To enable this evaluation, the original implementation of the Struck tracker was updated for compatibility with both the OpenCV 4.9 library and the ALOV300++ benchmark dataset.

The workstation-based evaluation of these algorithms, as reported in this study, provides a foundational basis for ongoing research into their feasibility and performance on resource-constrained SoC-FPGA platforms, such as the Xilinx ZC706 development board.

The remainder of this document is structured as follows: Section 2 presents a brief review of related work. Section 3 outlines the core principles of the Struck tracker and the characteristics of the ALOV300++ dataset. Section 4 provides a detailed explanation of the methodology employed in this study. Section 5 presents and discusses the results of the conducted experiments. Finally, Section 6 outlines the conclusions and future directions of this review.

2 Related Work

Several studies have explored the evaluation and enhancement of the Struck algorithm in comparison with traditional and modern tracking paradigms, including tracking-by-detection, correlation filter-based approaches, and recent deep learning models, using diverse benchmark datasets and feature representations.

Adamo et al.[14] analyze the TLD (Tracking-Learning-Detection) and Struck algorithms, originally developed using Fern and Haar visual features, respectively. Their study aims to evaluate the performance of these trackers when standard descriptors are replaced with local feature representations, including Local Binary Patterns (LBP), Local Gradient Patterns (LGP), and Histogram of Oriented Gradients (HOG). The authors observe that Struck's structured SVM, enhanced by kernel mapping, performs particularly well with LBP, LGP, and HOG, thereby improving its adaptability in dynamic tracking scenarios. The comparative experiments were conducted using the PETS2009 dataset, a benchmark commonly used for multi-camera tracking. The results on PETS2009 suggest that Struck, when equipped with robust local descriptors, generalizes effectively—and could potentially benefit even further when evaluated on larger and more challenging datasets such as ALOV300++.

Wang et al.[15] established theoretical connections between two state-of-the-art correlation filter (CF) trackers—Spatial Regularization Discriminative Correlation Filter (SRDCF) and Correlation Filter with Limited Boundaries (CFLB)—as well as the structured output tracker Struck. While the theoretical aspects of their work fall outside the scope of this study, it is important to note that their findings were supported by extensive experiments using the OTB50 and OTB100 benchmark datasets. Nevertheless, alternative benchmarks such as ALOV300++ offer a complementary evaluation environment with greater category diversity and complexity, making them particularly valuable for assessing tracking performance in more diverse and realistic visual scenarios.

Finally, with respect to the performance of Struck in comparison to more recent deep learning-based trackers, Minimol et al.[16] conducted comprehensive benchmarking across three widely recognized datasets: ALOV300++, OTB (Online Tracking Benchmark), and VOT Challenge. In their study, Struck was employed as a classical non-deep learning baseline due to its well-established reliability and structured SVM-based

framework. The evaluation was carried out using standard performance metrics, including the F-score, One-Pass Evaluation (OPE), robustness, and accuracy. It is relevant to highlight that, while the results demonstrated that Guided MDNet significantly outperformed Struck across all major evaluation criteria—particularly in reducing failure rates and maintaining consistent target localization under appearance variations— this study is referenced solely for its contextual relevance regarding the use of Struck and the ALOV300++ dataset. The use of deep learning models for VOT systems falls outside the scope of the present work.

3 Materials and Methods

The current methodology was developed with the primary objective of studying the public-domain implementation of the Struck tracker by Hare et al.[10], in order to evaluate its potential for deployment on a workstation platform. This evaluation represents the first phase of a broader research effort, with a future stage dedicated to implementing and optimizing the tracker on a resource-constrained SoC-FPGA platform.

The following main steps were undertaken as part of the reported methodology:

- Update of the Struck tracker (Section 3.1)
- Development of the ALOV300++ evaluation protocol (Section 3.2)
- Performance evaluation of Struck performance by category (Section 4.1)
- Comparative performance evaluation of studied trackers (Section 4.2)
- Overall performance analysis and discussion (Section 4.3)

3.1 Update of the Struck Tracker

The original Struck algorithm implementation was developed using OpenCV 2.4. For this study, the codebase was updated to ensure compatibility with OpenCV 4.9. This migration required a series of modifications due to significant changes in the OpenCV API across versions. The two most critical updates are detailed below:

a) Replacement of the IplImage structure with cv::Mat:. In OpenCV 2.4, image data was handled using the IplImage structure—a legacy C-style data type inherited from the Intel Image Processing Library. In contrast, OpenCV 4.9 uses cv::Mat, a modern C++ class that offers superior memory management, object-oriented design, and improved compatibility with contemporary OpenCV functions. Given that IplImage is now deprecated, all instances of it in the source code were replaced with cv::Mat, and all associated image-handling routines were updated accordingly.
b) Update of core OpenCV header file references. The original implementation included legacy headers such as opencv/cv.h and opencv/highgui.h, which are no longer supported in recent OpenCV versions. These were updated to their modern equivalents: opencv2/core/core_c.h and opencv2/highgui/highgui_c.h, respectively. These headers are essential for accessing OpenCV's core data structures and

GUI functionalities and ensure compatibility with OpenCV's modular architecture introduced in version 2.4 and formalized in later releases.

The update process also required configuring the CMake build environment to ensure compatibility with the updated OpenCV libraries. These configurations enabled successful compilation and execution of the tracker across different operating systems.

3.2 Implementation of the ALOV300++ Evaluation Protocol

A conventional one-pass evaluation (OPE) protocol was developed in Python to manage the workflow of dataset evaluation. The One-Pass Evaluation (OPE) protocol is the standard evaluation strategy employed in ALOV300++ to assess tracker performance on a per-sequence basis without reinitialization. This approach simulates a real-world tracking scenario in which the tracker must operate autonomously, without any manual intervention.

The following steps outline the development process for implementing the standard One-Pass Evaluation (OPE) protocol for the Struck tracker. Once the procedure was established, only minimal modifications were required to adapt the protocol for the other trackers evaluated in this study.

a) Preliminary analysis of the original struck evaluation procedure: A preliminary review of the original Struck implementation was carried out to understand its evaluation procedure with the OTB dataset. The main observation focused on how annotation files were processed to compute the Intersection over Union (IoU) metric. The OTB dataset provides ground truth annotations on a frame-by-frame basis, enabling straightforward calculation of per-frame IoU. In contrast, the ALOV300++ dataset includes annotations at fixed intervals, which vary across sequences, thereby requiring additional preprocessing to align predictions and annotations for consistent evaluation.
b) Analysis of ground truth structure and distribution in the ALOV300++ dataset: A detailed analysis was conducted on all 314 sequences of this dataset to assess the structure and distribution of the ground truth annotations. It was found that 39 sequences use a different annotation sampling interval—specifically, annotations spaced every five frames—consistent with what is reported by the original dataset authors. This irregularity further emphasizes the need for standardized preprocessing to ensure uniform metric computation across all sequences.
c) Standardization of ground truth format: All annotation files were reviewed and, when necessary, converted to the widely adopted (Xmin, Ymin, Width, Height) format to ensure compatibility and consistency across evaluation routines.
d) Definition of evaluation sequences file: A control CSV file was generated to specify the subset of sequences to be analyzed, including the initial and final frame indices for each sequence.
e) Tracker Execution Automation: A Python script (OPEtest.py) was developed to automate the execution of the tracker across the sequences and frame ranges specified in the control CSV file. For each evaluated sequence, the script generates two output files: one containing the bounding boxes produced by the tracker, and another recording the processing time required to generate each output per frame.

f) Post-experiment performance analysis: A second Python script (CategoryAnalysis.py) was developed to compute the performance metrics—F1-score, precision, and frames per second (FPS)—for all trackers and sequences specified in OPEtest.py. In addition to numerical evaluation, the script also generates the corresponding plots for visual analysis.

Although the implementation supports evaluation on any subset of sequences, this study considered the complete set of sequences across all categories for each of the aforementioned trackers.

3.3 The Struck Tracker

The publication "Struck: Structured Output Tracking with Kernels" by Sam Hare et al.[10] represents a seminal contribution to the field of VOT. This work introduced an adaptive tracking-by-detection framework based on structured output prediction, employing a kernelized structured output support vector machine (SVM) to directly model the relationship between candidate image regions and their corresponding object locations[17]. By reformulating visual tracking as a structured output prediction problem, the authors eliminate the need for an explicit intermediate classification step—such as training a separate binary classifier—thereby achieving a more integrated, end-to-end framework that is both theoretically robust and practically effective.

The key properties of the Struck algorithm include (see Figure 1):

a) the use of an SVM-based structured predictor to capture the contextual relationship between the target and its surroundings.
b) the application of kernel methods to address non-linearities in object representation.
c) a budgeting mechanism that constrains the growth of support vectors, enhancing computational efficiency during online operation.

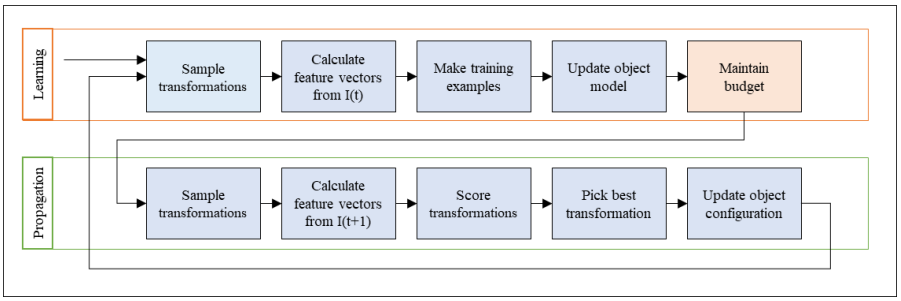

Fig. 1. Functional representation of the Struck tracker (modified from [10]).

Struck's main advantages include its online learning capability for real-time adaptation, computational efficiency through support vector budgeting, and robustness to appearance variations like illumination changes and deformation[10]. However, it is sensitive to partial occlusions, as it continues to update its model based on potentially corrupted inputs. This can lead to a well-known issue in tracking called drift, where the

model gradually shifts away from the true target due to erroneous updates over time[18].

3.4 The ALOV300++ Benchmark Dataset

The Amsterdam Library of Ordinary Videos (ALOV300++) benchmark dataset, introduced by Smeulders et al.[1], consists of 314 short video sequences organized into 14 challenge categories, primarily sourced from real-world YouTube footage. Designed to reflect diverse visual conditions—such as occlusion, motion blur, illumination changes, and background clutter—the dataset facilitates detailed evaluation of tracker performance under specific challenges. Its realism is reinforced by the inclusion of authentic video artifacts like compression noise and dynamic lighting variations. ALOV300++ comprises a large number of annotated frames, with ground-truth bounding boxes provided every fifth frame for most sequences, and is consistent with other standard benchmarks in the field [19].

The short sequences average 9.2 seconds in length, with a maximum of 35 seconds, and an additional category includes ten longer videos ranging from one to two minutes, enhancing its temporal coverage and evaluation flexibility. Table 1 presents a structured overview of the ALOV300++ dataset, detailing the number of video sequences, available raw frames, and annotated frames per category. The dataset comprises 314 video sequences, yielding a total of 151,657 raw frames and 16,337 annotations.

Table 1. The ALOV300++ dataset.

Category	Sequences	Available raw frames	Annotations
01-Light	33	17,789	1,321
02-SurfaceCover	15	7,118	638
03-Specularity	18	6,960	916
04-Transparency	20	5,284	815
05-Shape	24	10,801	1,133
06-MotionSmoothness	22	10,546	636
07-MotionCoherence	12	8,734	409
08-Clutter	15	8,013	696
09-Confusion	37	11,554	1,178
10-LowContrast	23	7,675	1,036
11-Occlusion	34	13,442	1,371
12-MovingCamera	22	8,154	1,025
13-ZoomingCamera	29	8,692	1,168
14-LongDuration	10	26,895	3,995
Total	314	151,657	16,337

3.5 Evaluation Metrics

Following, metrics used in this work are indicated: the frames per second (FPS) metric measures the number of frames a tracking system can process in one second. It

reflects the real-time performance or speed of the tracking algorithm. Higher FPS indicates better suitability for real-time applications, such as autonomous navigation or live surveillance.

IoU is a fundamental metric defined as the ratio of the area of intersection to the area of union of the predicted bounding box and the ground truth bounding box [20, 21]. This metric provides an intuitive measure of localization accuracy, with values ranging from 0 (no overlap) to 1 (perfect overlap). It is often used as a threshold criterion (usually defined at 0.5) to determine whether a detection is considered a true positive (see Figure 2).

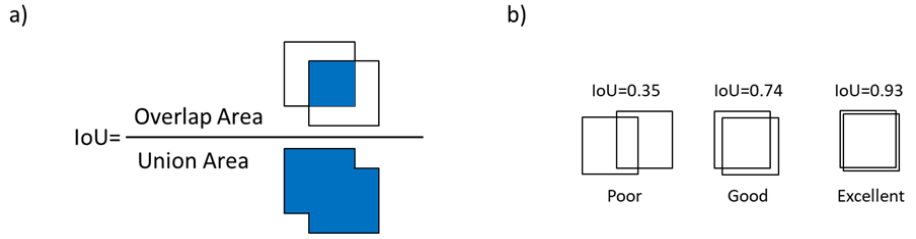

Fig. 2. Intersection over union (IoU) metric. a) IoU is calculated as the ratio between the intersection area and the union area of the ground truth and the detected bounding box. b)IoU examples (image modified from [21]).

Precision, Recall and F1-score are related metrics. Precision quantifies the proportion of predicted bounding boxes that are correct, while recall quantifies the proportion of ground truth objects that are successfully detected. The F1-score combines both precision and recall into a single metric by computing their harmonic mean, providing a balanced assessment of detection performance, especially when there is an uneven trade-off between false positives and false negatives. In the context of the ALOV300++ dataset, recall is typically considered to be 1, as each sequence involves tracking a single object with continuous ground truth annotations, and standard evaluation protocols assume that the tracker outputs predictions for all frames, eliminating the presence of false negatives[1].

4 Experiments and Results

A series of One-Pass Evaluation (OPE) experiments were conducted to assess the performance of the Struck tracker in comparison to the CSRT, KCF, and MIL trackers. The experiments were executed on a Whitebox workstation equipped with an Intel Core i5 processor (3.50 GHz), 32 GB of RAM, and running the Ubuntu 22.04 operating system. For all experiments involving the Struck tracker, the default configuration file provided with the original implementation was used.

4.1 Struck Performance Evaluation

For this experiment, the Struck tracker was executed on all 314 sequences of the ALOV300++ dataset. Precision and F1-score were evaluated on a per-category basis, with recall assumed to be 1.0 in all cases. The evaluation of the Struck tracker on the complete ALOV300++ dataset yielded a global precision of 0.650 ± 0.134 and an F1-score of 0.722 ± 0.118, indicating a generally robust performance across diverse tracking scenarios.

Notably, the highest tracking accuracy was achieved in the 08-Clutter category, where the tracker attained a precision of 0.905 ± 0.212 and an F1-score of 0.932 ± 0.162, suggesting strong resilience to background clutter and visual distractions. Conversely, the lowest performance was observed in the 14-LongDuration category, with a precision of 0.394 ± 0.395 and an F1-score of 0.461 ± 0.375, highlighting the tracker's limitations in maintaining reliable long-term tracking without drift (see Figure 3).

4.2 Evaluation of Trackers by Category

For comparative purposes, the previous experiment was repeated with CSRT, KCF, and MIL trackers, in addition to a new execution of the updated Struck tracker. The primary objective was to evaluate and compare the tracking throughput of each algorithm under identical conditions. To ensure consistency with the ALOV300++ dataset processing pipeline, the implementations of the selected trackers were adapted from official OpenCV examples.

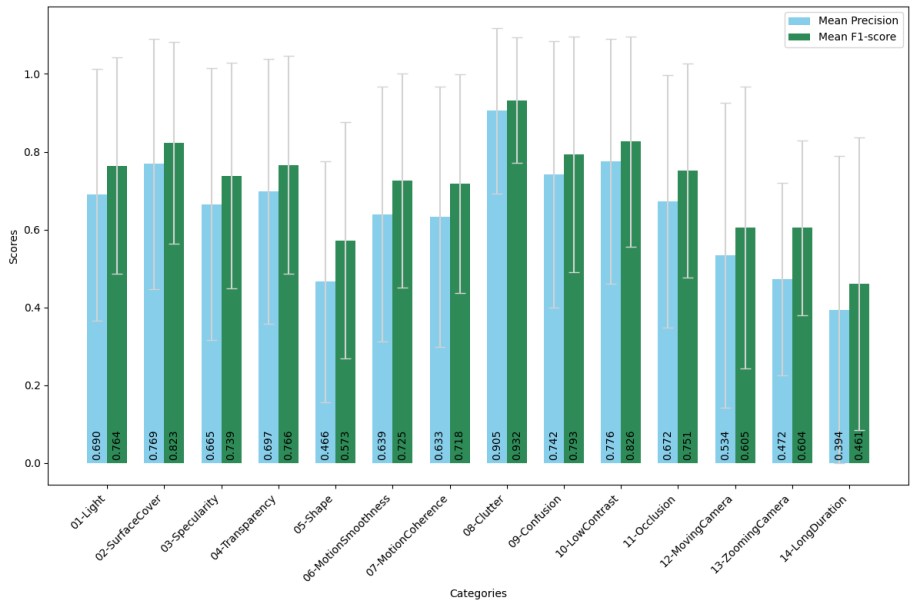

Fig. 3. Struck performance evaluation for all ALOV300++ categories.

F1-score analysis by category for the mentioned trackers are presented in Figure 4. Results show that Struck and CSRT generally achieve higher F1-scores across most challenge categories, with Struck performing particularly well in "02-SurfaceCover", "08-Clutter", "10-LowContrast", and CSRT excelling in "01-Light", "08-Clutter", and "10-LowContrast". MIL demonstrates moderately competitive performance in several categories, especially "02-SurfaceCover" and "08-Clutter" but struggles in "12-MovingCamera" and "14-LongDuration". KCF consistently records the lowest F1-scores across most categories, with notably poor results in dynamic camera scenarios such as "12-MovingCamera", "13-ZoomingCamera", and "14-LongDuration". Overall, Struck and CSRT exhibit stronger robustness under varied visual tracking challenges, particularly in complex or cluttered environments.

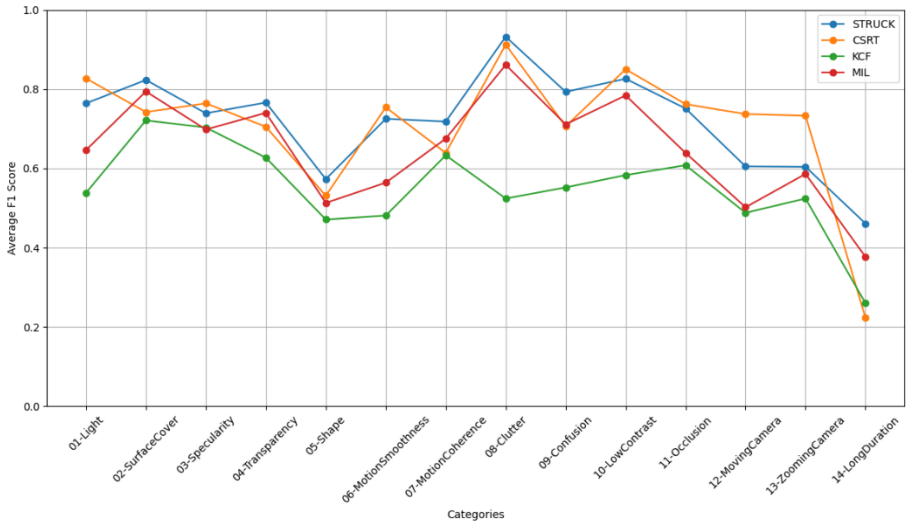

Fig. 4. Average F1-score of each category.

Regarding FPS performance across the 14 visual tracking challenge categories (see Figure 5), KCF consistently demonstrates the highest throughput, achieving over 1000 FPS in dynamic conditions such as "07-MotionCoherence" and "14-LongDuration" peaking at 300 FPS in the last category. CSRT offers a balanced compromise between speed and accuracy, with FPS values ranging from approximately 84.5 to 319.6, showing strong performance particularly in "14-LongDuration" and "01-Light". Struck and MIL, while significantly slower than KCF and CSRT, maintain stable frame rates between 42.2 and 86.4 FPS. Notably, Struck shows consistent FPS across categories, reflecting stable computational behavior. Overall, KCF leads in execution speed but at the expense of accuracy, while CSRT strikes a favorable trade-off for scenarios requiring both real-time performance and reliability.

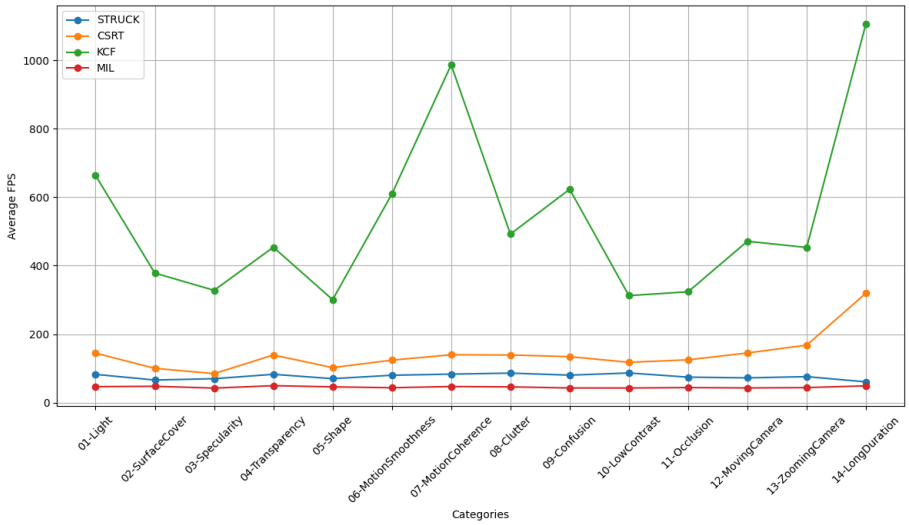

Fig. 5. Average frames per second of each tracker by category.

4.3 Global Results

The global results presented in Table 2 and Figure 6 indicate that Struck and CSRT exhibit relatively higher F1-scores—0.720 ± 0.122 and 0.706 ± 0.166, respectively—compared to KCF (0.551 ± 0.115) and MIL (0.649 ± 0.132), suggesting potentially better tracking performance under the tested conditions.

Table 2. Performance comparison of evaluated trackers.

Tracker	Precision	F1-score	FPS	Total dataset execution time
CSRT	0.633 ± 0.179	0.706 ± 0.166	135.900 ± 98.160	17 min, 49.77 sec
KCF	0.475 ± 0.115	0.551 ± 0.115	503.850 ± 583.050	9 min, 16.57 sec
MIL	0.566 ± 0.138	0.649 ± 0.132	44.780 ± 6.580	32 min, 55.78 sec
STRUCK	0.647 ± 0.139	0.720 ± 0.122	77.100 ± 20.110	23 min, 31.02 sec

However, the performance differences between Struck, CSRT, and MIL are moderate, and the overlapping standard deviations imply that additional statistical analysis would be required to determine whether these differences are statistically significant.

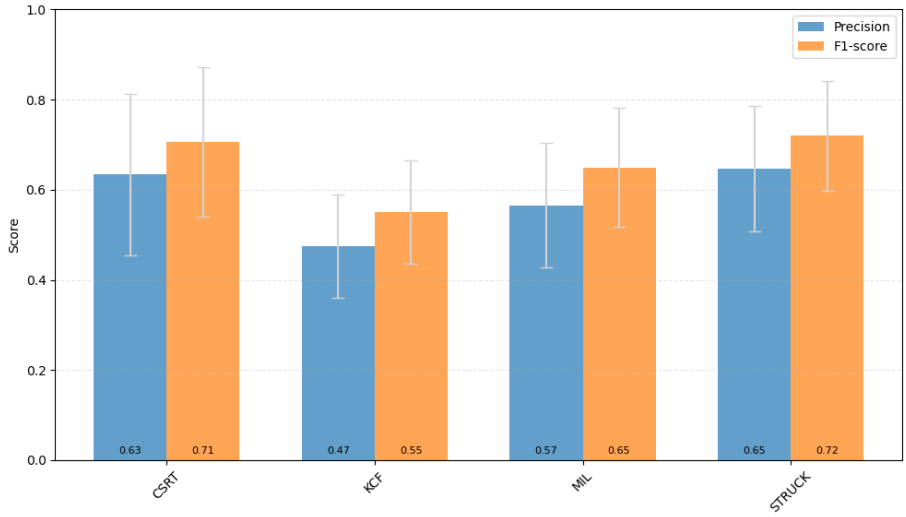

Fig. 6. Global precision and F1-score per tracker.

5 Conclusions

This work presents the implementation update of the original Struck algorithm by Hare et al., migrating from OpenCV 2.4 to OpenCV 4.9.0, and its comparative evaluation using the ALOV300++ dataset on a conventional workstation. Although the updated implementation was successfully tested on both Windows 10 and Ubuntu 22.04, this report includes only the experiments conducted under the Ubuntu environment. Extensive testing of the Struck algorithm's performance in terms of F1-score, precision, and frames per second (FPS) metrics was conducted on both mentioned operating systems using the ALOV300++ dataset. These evaluations confirmed the algorithm's efficiency and accuracy in various scenarios. It is worth mentioning that the Recall metric was not used because, due to the nature of the visual tracking task, it is theoretically always equal to 1. Implementations of CSRT, MIL, and KCF trackers were adapted to the ALOV300++ dataset and tested successfully on all 314 sequences.

The comparative analysis of F1-score and FPS across 14 challenging visual tracking categories reveals a clear trade-off between accuracy and speed among the evaluated trackers. Struck and CSRT consistently deliver high F1-scores, indicating robust tracking performance under varying visual conditions, with Struck slightly outperforming CSRT in cluttered and low-contrast scenarios. However, although KCF achieves the highest FPS across all categories, it demonstrates significantly lower accuracy particularly in complex or long-duration sequences. The observed FPS values confirm that MIL offers a moderate balance between tracking accuracy and processing speed but does not lead in either category. CSRT emerges as the most balanced option, offering a strong compromise between precision and computational efficiency. Struck remains a reliable choice for scenarios that prioritize accuracy, albeit at a higher computational cost. In contrast, KCF significantly outperforms the other trackers in terms of speed,

making it the preferred solution for real-time applications where processing throughput is more critical than precise target localization.

Future work will focus on the use of VOT algorithms and metaheuristics to design, implement, and evaluate a VOT system on a SoC-FPGA (System-on-Chip Field-Programmable Gate Array) platform. SoC-FPGA is a heterogeneous hardware architecture that integrates a conventional CPU processor core with FPGA fabric, enabling efficient hardware–software co-design. Building upon the current progress of this Ph.D. research, the next phase will involve the development and deployment of the updated Struck tracker on a SoC-FPGA platform, such as Xilinx's ZC706 development board, aiming to achieve real-time performance and resource-efficient implementation.

References

1. Smeulders, A.W.M., Chu, D.M., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.: Visual Tracking: An Experimental Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 36, 1442–1468 (2014). https://doi.org/10.1109/TPAMI.2013.230
2. Kristan, M.: The Tenth Visual Object Tracking VOT2022 Challenge Results :: ViCoS Prints, https://prints.vicos.si/publications/416/the-tenth-visual-object-tracking-vot2022-challenge-results, last accessed 2023/06/06
3. Soleimanitaleb, Z., Keyvanrad, M.A.: Single Object Tracking: A Survey of Methods, Datasets, and Evaluation Metrics. ArXiv. (2022)
4. Chamberlain, R.D.: Architecturally truly diverse systems: A review. Future Generation Computer Systems. 110, 33–44 (2020). https://doi.org/10.1016/j.future.2020.03.061
5. Méndez López, V.A., Soubervielle Montalvo, C., Núñez Varela, A.S., Pérez Cham, O.E., González Galván, E.J.: A Review of Design Methodologies and Evaluation Techniques for FPGA-Based Visual Object Tracking Systems. International Journal of Combinatorial Optimization Problems and Informatics. 15, 127–145 (2024). https://doi.org/10.61467/2007.1558.2024.v15i5.571
6. Abbass, M.Y., Kwon, K.-C., Kim, N., Abdelwahab, S.A., El-Samie, F.E.A., Khalaf, A.A.M.: A survey on online learning for visual tracking. Vis Comput. 37, 993–1014 (2021). https://doi.org/10.1007/s00371-020-01848-y
7. Park, E., Berg, A.C.: Meta-tracker: Fast and Robust Online Adaptation for Visual Object Trackers. 11207, 587–604 (2018). https://doi.org/10.1007/978-3-030-01219-9_35
8. Lu, J., Hoi, S.C.H., Wang, J., Zhao, P., Liu, Z.-Y.: Large Scale Online Kernel Learning. (2016)
9. Hoi, S.C.H., Sahoo, D., Lu, J., Zhao, P.: Online learning: A comprehensive survey. Neurocomputing. 459, 249–289 (2021). https://doi.org/10.1016/j.neucom.2021.04.112
10. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: 2011 International Conference on Computer Vision. pp. 263–270 (2011). https://doi.org/10.1109/ICCV.2011.6126251
11. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-Speed Tracking with Kernelized Correlation Filters. IEEE Transactions on Pattern Analysis and Machine Intelligence. 37, 583–596 (2015). https://doi.org/10.1109/TPAMI.2014.2345390

12. Babenko, B., Yang, M.-H., Belongie, S.: Visual tracking with online Multiple Instance Learning. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 983–990 (2009). https://doi.org/10.1109/CVPR.2009.5206737
13. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C. (eds.) Computer Vision – ECCV 2012. pp. 702–715. Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33765-9_50
14. Adamo, F., Carcagnì, P., Mazzeo, P.L., Distante, C., Spagnolo, P.: TLD and Struck: A Feature Descriptors Comparative Study. In: Mazzeo, P.L., Spagnolo, P., and Moeslund, T.B. (eds.) Activity Monitoring by Multiple Distributed Sensing. pp. 52–63. Springer International Publishing, Cham (2014). https://doi.org/10.1007/978-3-319-13323-2_5
15. Wang, J., Zheng, L., Tang, M., Feng, J.: A Comparison of Correlation Filter-Based Trackers and Struck Trackers. IEEE Transactions on Circuits and Systems for Video Technology. 30, 3106–3118 (2020). https://doi.org/10.1109/TCSVT.2019.2931924
16. Venugopal Minimol, P., Mishra, D., Gorthi, R.K.S.S.: Guided MDNet tracker with guided samples. Vis. Comput. 38, 1135–1149 (2022). https://doi.org/10.1007/s00371-021-02072-y.
17. Fiaz, M., Mahmood, A., Javed, S., Jung, S.K.: Handcrafted and Deep Trackers: Recent Visual Object Tracking Approaches and Trends. ACM Comput. Surv. 52, 43:1-43:44 (2019). https://doi.org/10.1145/3309665
18. Ning, J., Yang, J., Jiang, S., Zhang, L., Yang, M.-H.: Object Tracking via Dual Linear Structured SVM and Explicit Feature Map. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4266–4274. IEEE, Las Vegas, NV, USA (2016). https://doi.org/10.1109/CVPR.2016.462
19. Soleimanitaleb, Z., Keyvanrad, M.A.: Single Object Tracking: A Survey of Methods, Datasets, and Evaluation Metrics. arXiv:2201.13066. (2022)
20. Trigka, M., Dritsas, E.: A Comprehensive Survey of Machine Learning Techniques and Models for Object Detection. Sensors. 25, 214 (2025). https://doi.org/10.3390/s25010214
21. Terven, J., Cordova-Esparza, D.M., Ramirez-Pedraza, A., Chavez-Urbiola, E.A., Romero-Gonzalez, J.A.: Loss Functions and Metrics in Deep Learning. Artif Intell Rev. 58, 195 (2025). https://doi.org/10.1007/s10462-025-11198-7