ISSN 1870-4069

Parallel Video Tracking System based on Honeybee Swarm Behavior and Evaluated on a Benchmark Dataset

Juan Alvaro de la Rosa-Ipiña, Carlos Soubervielle-Montalvo, Cesar Puente, Ruth Mariela Aguilar-Ponce, Luis Javier Ontañon García-Pimentel

Universidad Autónoma de San Luis Potosi, Faculty of Engineering, San Luis Potosi, México
alvarodelarosaip@gmail.com,
{carlos.soubervielle,cesar.puente}@uaslp.mx,
rmariela@fciencias.uaslp.mx,luis.ontanon@uaslp.mx

Abstract. This research aims to enhance the parallel implementation of the Zero-mean Normalized Cross-Correlation (ZNCC) algorithm for video tracking, with the goal of increasing processing speed without compromising accuracy. Speed improvements are anticipated through the integration of the bio-inspired Honeybee Search Algorithm (HSA) as an exploratory strategy within the tracking framework, using the improved ZNCC as the fitness function. The HSA algorithm has been adapted for video tracking by incorporating population management during the exploration phase, as well as the recruitment and harvesting processes. The completed tasks form the basis for implementing the parallel video tracking system more efficiently using the heterogeneous CPU-GPU architecture.

Keywords: Object Tracking, Honeybee Search Algorithm, Swarm Intelligence, Parallel Computing, Graphics Processing Unit.

1 Introduction

Video object tracking is the task of identifying a specific object across the sequence of images (or frames) that comprise a video [1], and is one of the most requested tasks in the computer vision area due to its multiple application field with unlimited potential such as medical, military and entertainment applications, among others.

For years, multiple video tracking algorithms and methodologies have been proposed to achieve this task with precision. Video tracking is a task that consists of identifying the position of an object in each frame of a video sequence [1]. However, the analysis of video frames is an exhaustive and delicate task that has not been perfected yet, in addition to entailing a great demand for time and computational resources. For this reason, efforts have been made to enhance these systems through different software and hardware resources [1,2,3,4].

Perfecting object tracking methods is important due to the potential catastrophic failures that these could generate when applied to critical systems and especially to systems where the integrity of human lives is involved, such as medical, military, or certain everyday use systems, such as autonomous driving vehicles. These failures may be due

to the susceptibility of video tracking algorithms to certain obstacles commonly present in video sequences, in addition to the tracking algorithms' large computational costs and complexity that cause considerably serious delays in response times.

Video object tracking remains a challenging problem without a definitive solution. Although various approaches—such as Zero-mean Normalized Cross-Correlation (ZNCC) [2]—have been proposed, they still leave room for improvement in areas such as energy efficiency, computational resource demands, and robustness against noise, among other factors.

2 Related Work

To solve the video tracking problem, different solutions have been proposed, with the use of bioinspired metaheuristics being of special interest for this research. Algorithms inspired by nature have shown great potential as video tracking methods increasing the processing speed by reducing their resources and computational operations. Some of the most popular bioinspired metaheuristics in recent years in the video tracking research context are ABC (Artificial Bee Colony) [3], BA (Bat Algorithm) [4], and especially PSO (Particle Swarm Optimization) [5] which is one of the most used metaheuristic algorithms and has several studies and papers published about how it can be used as an optimizing algorithm for video tracking.

Another bioinspired metaheuristic is HSA (Honeybee Search Algorithm) which was proposed [2] to be used along the ZNCC video tracking algorithm as the central fitness function. An implementation that demonstrated promising results (with approximately a 30% accuracy and 13.8 FPS [6]) but with some areas of opportunity such as the FPS rate which is considerably low in this algorithm applied to video tracking and leads to deeper investigations and the proposal of this proposal. This approach is one that has not been touched on literature until now except for the cited works [2,6,7,8].

Other implementations of both the ZNCC algorithm and its original variant, NCC, have been investigated, as is the case of the initial evaluations that were made of this algorithm on the ALOV300++ dataset [9] where the NCC was shown to have an accuracy of 57%. In these tests, the Struck algorithm proved to have the highest accuracy of 66%.

In [2] it was proposed to use the HSA algorithm along ZNCC method as a possible solution to the optimization problem in video tracking problem. Tests have shown that HSA in conjunction with NCC/ZNCC can be effective and useful in a video tracking scenario. However, this implementation still requires to be perfected and tested to find the optimal conditions under which both algorithms can work more efficiently and selecting which combination of ZNCC-HSA in the video tracking obstacles works better. Even surpassing previously obtained measurements of 30% accuracy and 13.8 FPS [2,6].

Among the algorithms with which this implementation is compared, Struck and SiamMask stand out being Struck one of the most popular and well received online video tracking algorithms with the best results in accuracy, while SiamMask is perhaps the most popular deep learning algorithm in the video tracking research area [10]. Against

these two trackers, the NCC – HSA implementation obtained lower but also promising results that indicated that after applying certain improvements it could be capable of obtaining comparable or even superior results.

3 Methodology

In this work, we propose the use of the evolutionary metaheuristic known as Honeybee Search Algorithm (HSA) to estimate the most probable positions of the target object within each frame. In the HSA context, each individual (bee) is assigned to a specific position (pixel) in the frame, and its suitability as the object's location is evaluated using the Zero Mean Normalized Cross-Correlation (ZNCC) image similarity metric. To guide the population of individuals toward the most promising regions of the frame, the algorithm executes a three-phase evolutionary process designed to iteratively generate and refine individuals in increasingly accurate positions.

To accelerate the aforementioned video tracking process, a heterogeneous CPU-GPU architecture is proposed. In this scheme, the decision-making processes of the HSA are executed on the CPU, while the evaluation of its fitness function—based on the ZNCC algorithm—is offloaded to the GPU. A more detailed description of this parallel architecture is provided in a later section.

3.1 ZNCC (Zero Mean Normalized Cross-Correlation)

Zero-mean Normalized Cross-Correlation (ZNCC) is a widely used similarity measure in template matching and video object tracking. It compares a reference template to regions in a target image to identify the best match, while remaining robust to variations in brightness and contrast. The ZNCC value is computed using the following equation:

$$\gamma(u,v) = \frac{\sum_{x,y}\left[I(x,y)-\bar{I}_{u,v}\right]\left[t(x-u,y-v)-\bar{t}\right]}{\sqrt{\sum_{x,y}\left[I(x,y)-\bar{I}_{u,v}\right]^2 \sum_{x,y}\left[t(x-u,y-v)-t\right]^2}}. \quad (1)$$

Where (u, v) are the coordinates of the analysis areas top left corner, I(x, y) refer to all the pixels that compose the analysis area, $\bar{t}$ is the average template value and $\bar{I}$ is the a value obtained through the sum of pixel values in an area and divided into the size of that particular area (m x n). Particularly, $\bar{t}$ and $\bar{I}$ are defined as follows:

$$\bar{t} = \frac{\sum_{x,y} t(x-u,y-v)}{m \times n}. \quad (2)$$

$$\bar{I}_{u,v} = \frac{\sum_{x,y} I(x,y)}{m \times n}. \quad (3)$$

3.2 HSA (Honeybee Search Algorithm)

The HSA algorithm is inspired by how bees search and collect food to find solutions in the most efficient possible way considering three phases which are applied to the video tracking in the following sense (see Fig. 1):

1. Exploration: the algorithm releases a group of "bees" generated by an evolutionary strategy algorithm to find the position(s) where the object is most likely to be located within the frame's search area. The ZNCC method is used to evaluate possible positions.
2. Recruitment: new bees are assigned to the areas with the best ZNCC evaluation. This is the only phase that does not use the evolutionary strategy algorithm.
3. Harvesting: the bees assigned in the previous phase go to most of the areas where the object must be located. This phase requires more computational power on the evolutionary strategy algorithm than the exploration phase since the most refined search for the object will be concentrated here.

To enhance the evaluation of candidate positions during both the exploration and harvesting phases, the ZNCC method will be optimized through thread-based parallel computing using the OpenCL library for Python. In this parallel computing environment, each bee will be assigned to an individual thread, where it will independently execute the ZNCC method to evaluate its position. This parallelization strategy is aimed at accelerating the computation of the fitness function in the HSA algorithm, which is defined by the ZNCC metric.

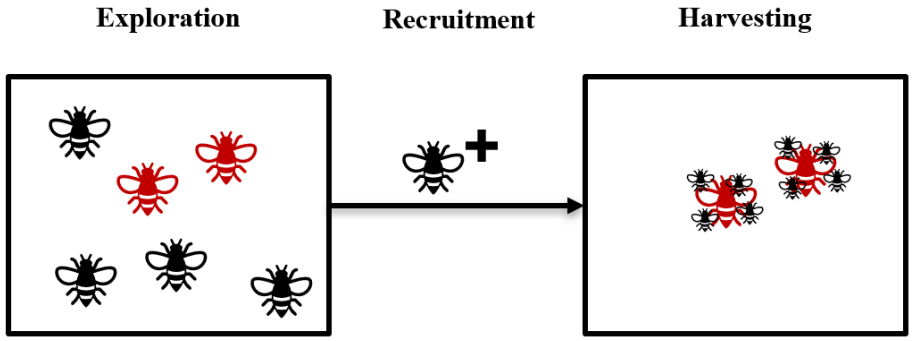

Fig. 1. HSA applied to video frame analysis. Bees look for the best positions (exploration), new bees are assigned to those positions (recruitment) and then they exploit them (harvesting).

3.3 ALOV300++ Dataset

To evaluate the system's performance, the ALOV300++ dataset [9] will be used. This is a dataset composed of more than 300 videos divided into different categories according to the properties and obstacles they present such as occlusion, reflection, shape changes, long duration, etc. Since it was proposed in 2014, it has become very popular in video tracking research due to its diversity and easy access and even its use has been decreasing through time, it has been still used in recent years because of the previously reasons that were mentioned [2,12].

The ALOV300++ dataset represents a challenge for any video tracking system due to the robust diversity of its videos which allows us to observe in detail how a video tracker can have a better or worst perform under certain conditions and certain types of videos.

4 Use of GPU

The emergence of increasingly specialized hardware architectures and components designed to address specific computational tasks has driven the search for ways to harness their potential in current areas of computational research. An example of this is GPUs, which are an architecture designed specifically to process graphics tasks. They can serve as powerful tools, particularly in applications such as video tracking. In addition to their potential in video tracking algorithms (such as ZNCC), great potential has been observed in GPUs for optimization and parallelization algorithms such as HSA and PSO [2].

After the CPUs, the GPUs have been the most discussed hardware architecture in the video tracking field not only because of its potential in video and image processing, but also due to its possible use as a component for executing parallelization metaheuristics. However, the use of GPUs in this area still has some unexplored potential because the correct parameters that can take advantage of this architecture in the most efficient way possible have not been found yet [5,7]. Also, the GPUs' advantages over other architectures such as the CPU or FPGA have not been discovered with certainty yet neither [8].

To accelerate the entire ZNCC – HSA system (see Fig 2.), we seek to take advantage of GPUs, and their specific components designed for the graphic resources processing that can also be used to improve the parallel programming in HSA.

5 System Evaluation

The complete evaluation of the system is sought by searching and implementing the appropriate metrics such as F1-Score and FPS (Frames Per Second), as well as other specific measures used for the correct video tracking system evaluation.

FPS (Frames Per Second). The frames per second (FPS) measurement is what we are going to use to measure the speed of the system to evaluate how many frames the system can analyze in one second of execution and confirm whether or not the objective of the system reaching the 15 FPS.

F1-Score. To measure accuracy, we will use the measure known as F1-Score [11], also known as the Sørensen-Dice Coefficient, a variation of the F-Score metric that is defined by the following equation:

$$\mathrm{F} = (1 + \beta^2) \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{(\beta^2 \times \mathrm{Precision}) + \mathrm{Recall}}. \tag{4}$$

where β is a factor that determines the weight that Precision and Recall will have in the result. Precision and Recall are obtained through:

$$\mathrm{Precision} = \frac{TP}{TP+FP}. \tag{5}$$

$$\text{Recall} = \frac{TP}{TP+FN}. \tag{6}$$

where TP = True Positives, FP = False Positives and FN = False Negative, and these are obtained through the IoU (Intersection Over Union) measure.

F1-Score is a F-Score variant where β = 1, which gives equal weight to both Precision and Recall resulting in a balanced result where both metrics have the same weight. When we substitute the β value in the original F-Score equation we get the next F1-Score equation:

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{7}$$

6 Improvements

As mentioned in section 2, Dr. Oscar Perez-Cham proposed an original implementation of the HSA algorithm in a parallel approach applied to video object tracking [2]. This implementation was programmed in C code language.

A review and execution on this previous C code showed some areas of opportunity that could be taken advantage of to improve the implementation:

- Lack of modularity and flexibility in the code.
- Better use of the feedback provided by HSA.
- Taking advantage of the optimized and specialized libraries of a programming language such as Python that is higher level than C and C++.
- Better use of the GPU and CPU.

6.1 Modular Code Implementation

The development of a new version of the ZNCC - HSA implementation through a modular approach is expected to take advantage of the multi-paradigm property of Python language as well as the optimization of its specialized libraries to create an optimally flexible and easily manageable system.

The system is designed to focus on a core autonomous module (see Fig. 3) that would only be responsible for executing the video sequences frame by frame in addition to coordinating the work of the rest of the modules that would oversee more specific tasks such as the tracking module. which would oversee tracking the object itself and where the ZNCC code would be implemented. Similar modules or modules that are destinated to achieve the same function will be organized into packages which in addition to keeping the code organized, will allow an easy export and integration of those modules inside and outside the system.

Besides the core module, the only other independent module would be the testing module, as it is intended to be a system capable of running and calculating different metrics without the need to rely on the runtime analysis of the main module. Instead, this module could receive results files and analyze them after the execution of the main video tracking system to subsequently calculate metrics such as F1-Score. Still, this test

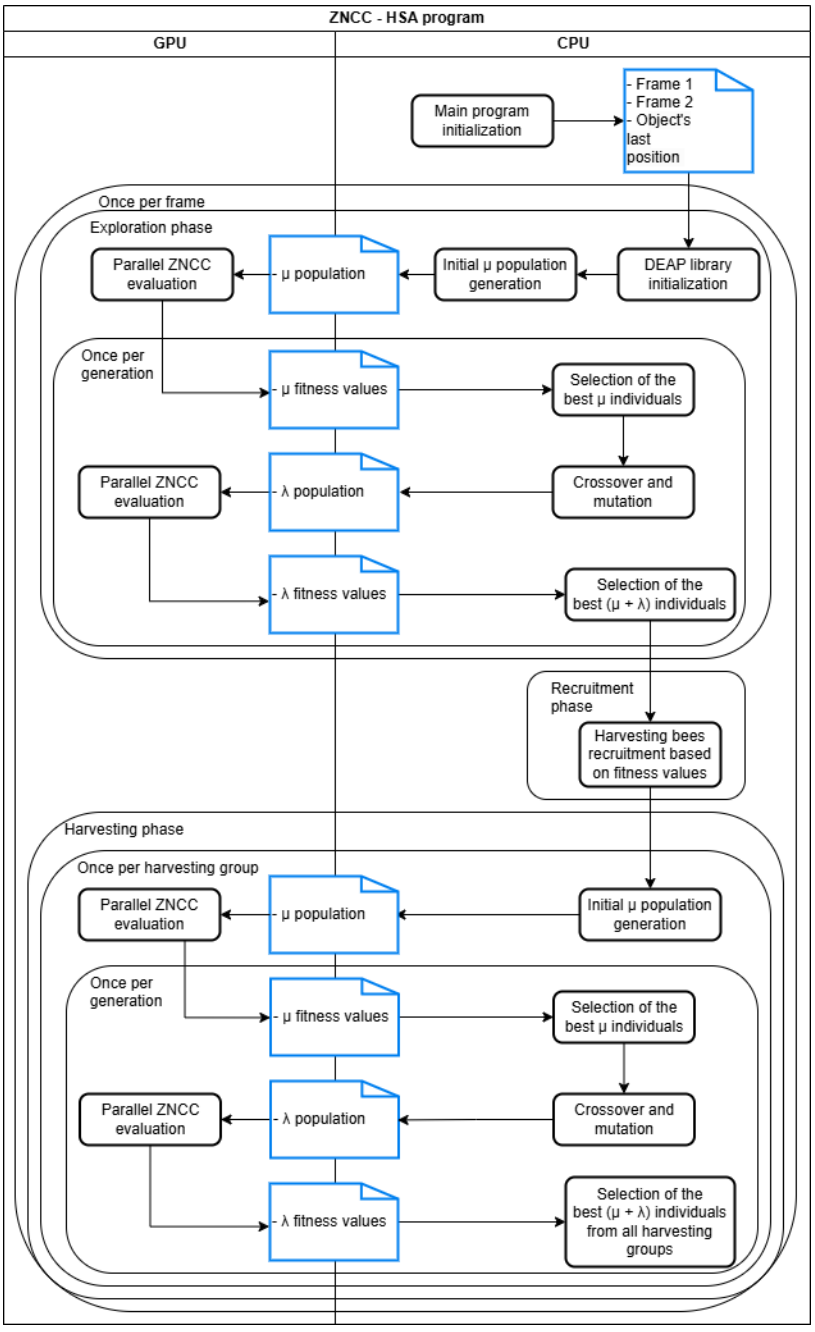

Fig. 2. Description of the task distribution on the proposed heterogeneous CPU-GPU architecture.

module could be integrated into the core module to calculate the necessary metrics at runtime.

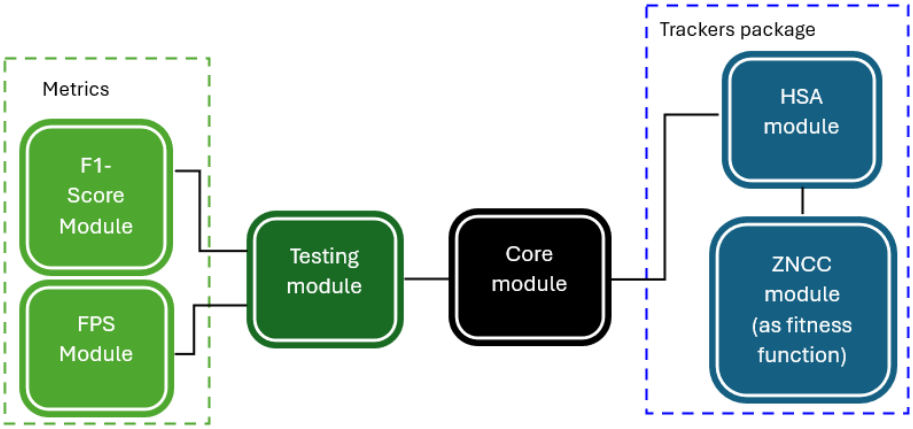

Fig. 3. Brief look at the current modular distribution of the system.

6.2 ZNCC Algorithm Improvements

As said, the overall accuracy of ZNCC performance in frame-by-frame analysis needs to be improved because it is still not 100% perfect, especially in certain types of videos that present some specifically difficult obstacles that cause the system to obtain a low accuracy percentage.

The ZNCC algorithm required improvements to increase its precision in the frame-by-frame video analysis since it has not reached 100% accuracy and in the case of some video types, it still obtained a quite low percentage. The purpose of these improvements is to reach approximately 62% precision, which is above the results obtained so far with the ZNCC algorithm [9].

Random coordinates. It was observed that the parallel execution of the ZNCC algorithm involved an exhaustive search using 100% of the frame's pixels, leading to redundancies when analyzing similar areas, which increased computational cost and caused issues like memory overflow especially with high-resolution videos.

Based on this observation, it was proposed that by analyzing less than 100% of the pixels, selected randomly and uniformly across the frame, redundancies could be reduced. According to the Pareto principle [13], an optimal percentage of random coordinates is estimated to be 20%.

Search area reduction. Based on the assumption that an object cannot undergo drastic displacement between consecutive video frames, the search area for object tracking was significantly reduced. Instead of scanning the entire frame, the algorithm now restricts the search to a localized region around the object's previous position where it is most likely to appear in the next frame. This optimization reduces

computational load by eliminating unnecessary searches and improves tracking accuracy by concentrating on the most probable location of the object [14].

The reduction of the search area is guided by four directional percentages—one for each side of the region—determined by the Honeybee Search Algorithm (HSA). These percentages indicate the likelihood of the object's movement in each direction, allowing the search to focus more heavily on the areas where the object is most likely to appear. To constrain the search space, the maximum search radius is limited to three times the object's size, with the directional percentages calculated relative to this radius (see Fig. 4).

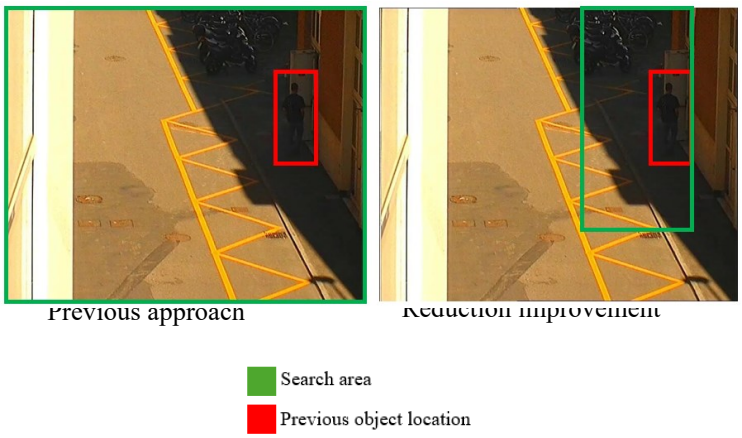

Fig. 4. Change on the search area with the new reduction improvement using four predetermined percentages based on 3 times the object size radius.

Blinds method. This is a variant of the pre-existent down-sampling method [15]. It is based on the idea that an object can still be distinguished even if there is low interference in the image. A change was implemented in the ZNCC algorithm code to the cycles that traverse the image pixels, so instead of moving pixel by pixel, a jump of two pixels is made and the system only process half of the frame's information, similar to watching the image pass through blinds (see Fig. 5)

Fig. 5. The idea behind the blinds method. a) Original frame, b) same frame after applying the blinds method.

Although the mathematical results of the ZNCC equation vary when removing half of the pixels from the analysis, when applying the same blinds algorithm to both frames (the current and the previous one), the statistical relationship between these two is maintained, something that has been corroborated in experiments and tests where it is observed that there is no noticeable decrease in the effectiveness of the ZNCC algorithm tracking the object when this method is applied.

6.3 HSA Improvements

The HSA algorithm Python implementation is currently in development. The purpose of using the HSA metaheuristic is to make the system as fast as possible (even trying to reach 15 fps) by taking advantage of the specific HSA heuristic properties combined with the ZNCC video tracking process. Once completed, the HSA algorithm is expected to decide where to follow the object taking advantage of the previously mentioned ZNCC improvements.

The HSA is going to work efficiently by using its three phases strategically: the exploration phase is applied only in the first frame to locate the object, while the recruitment and harvesting phases handle tracking in subsequent frames. The exploration phase is triggered again only if the object is lost, based on a predefined threshold—expected to be 62% precision (see Fig. 6).

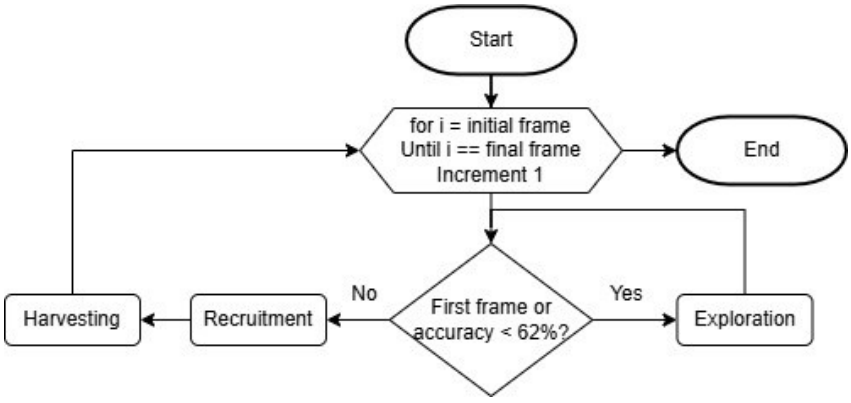

Fig. 6. General process of the HSA algorithm through frames and how its different phases are executed according to the current precision.

The previously explained usage of the frame-to-frame feedback will help not only increase the success possibility of finding the object in a frame but also make it faster.

7 Experiments and Results

The current implementation of the HSA algorithm with just the exploration phase has been tested on 33 videos coming from the 14 ALOV300++ dataset categories. These tests were performed to find the optimal mutation and crossover operators for the

algorithm considering HSA's original implementation was made to 3D reconstruction and its parameters were not changed in Dr. Perez-Cham's video tracking implementation [5].

Combinations of three crossover and mutation operators were implemented. The crossover operators that were selected are: uniform crossover, blend crossover and simulated binary crossover (SBX), while the mutation operators are: uniform mutation, polynomial mutation and gaussian mutation. The following graphs show the average results achieved on these experiments (see Fig. 7).

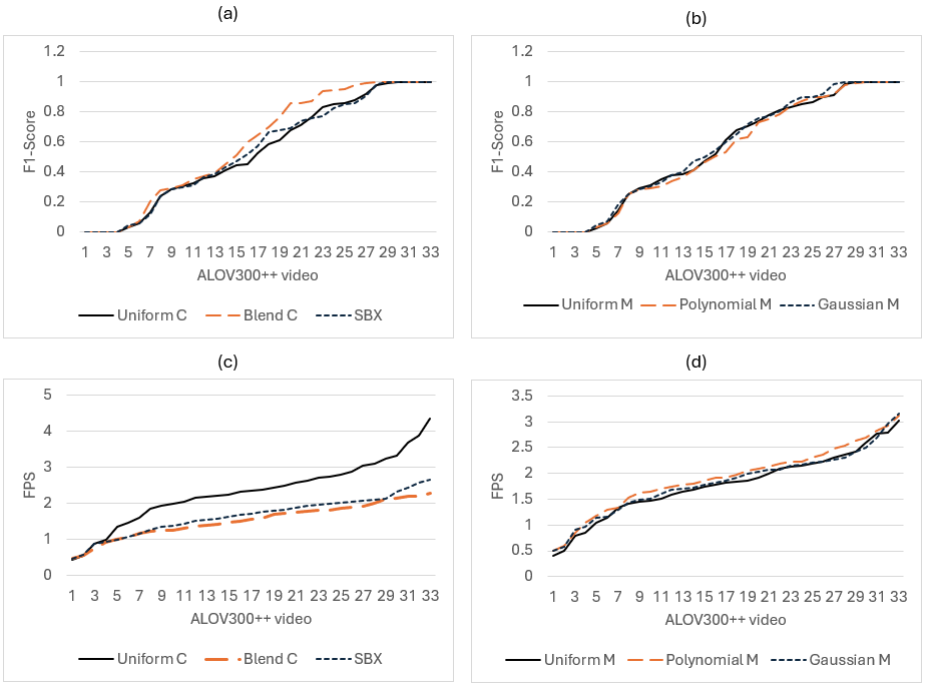

Fig. 7. a) F1-Score (accuracy) average behavior of the three crossover operators, b) F1-Score average behavior of the three mutation operators, c) FPS (speed) average behavior of the three crossover operators, d) FPS average behavior of the three mutation operators.

8 Conclusions

The final implementation is still under development; however, preliminary tests demonstrate significant improvements in both speed and accuracy compared to the original implementation presented in [5].

As mentioned above, the HSA is currently being fine-tuned to identify the most effective evolutionary operators. Based on the preliminary results, blend crossover and Gaussian mutation are currently considered the best candidates; however, further analysis is ongoing to confirm these findings.

Once the best operators are found, a basic version of the HSA algorithm will be available, in which optimization improvements will be implemented. In parallel, tests will be conducted on the entire ALOV300++ dataset.

The final version of HSA will incorporate all the improvements presented in this work; however, certain limitations remain. The most significant of these is the algorithm's limited robustness to changes in object size, which represents the main weakness of the current proposal. Addressing this limitation constitutes the primary direction for future work, along with enhancing the algorithm's resilience to object rotation, resilience to occlusion and improving computational efficiency. The latter may involve further code optimization or the development of a new implementation in C++ or another compiled language.

References

1. Maggio, E., Cavallaro, A.: Video Tracking: Theory and Practice. John Wiley & Sons (2011)
2. Chen, W.: Simulation of Multimedia Visual Image Motion Track Marking Based on Artificial Bee Colony Algorithm. Mathematical Problems in Engineering 1, 1–10 (2022). https://doi.org/10.1155/2022/8039589
3. Nenavath, H., Jatoth, R.K., Das, S.: A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. Swarm and Evolutionary Computation 43, 1–30 (2018). https://doi.org/10.1016/j.swevo.2018.02.011
4. Gao, M.L., Shen, J., Yin, L.J., Liu, W., Zou, G.F., Li, H.T., Fu, G.X.: A novel visual tracking method using bat algorithm. Neurocomputing 177, 612–619 (2016). https://doi.org/10.1016/j.neucom.2015.11.072
5. Perez-Cham, O.E., Puente, C., Soubervielle-Montalvo, C., Olague, G., Aguirre-Salado, C.A., Nuñez-Varela, A.S.: Parallelization of the honeybee search algorithm for object tracking. Applied Sciences 10(6), 2122 (2020). https://doi.org/10.3390/app10062122
6. Soubervielle-Montalvo, C., Perez-Cham, O.E., Puente, C., Gonzalez-Galvan, E.J., Olague, G., Aguirre-Salado, C.A., Cuevas-Tello, J.C., Ontanon-Garcia, L.J.: Design of a Low-Power Embedded System Based on a SoC-FPGA and the Honeybee Search Algorithm for Real-Time Video Tracking. Sensors 22(3), 1280 (2022). https://doi.org/10.3390/s22031280
7. Perez-Cham, O.E., Puente, C., Soubervielle-Montalvo, C., Olague, G., Castillo-Barrera, F.E., Nunez-Varela, J., Limon-Romero, J.: Automata design for honeybee search algorithm and its applications to 3D scene reconstruction and video tracking. Swarm and Evolutionary Computation 61, 100817 (2021). https://doi.org/10.1016/j.swevo.2020.100817
8. López, V.A.M., Montalvo, C.S., Varela, A.S.N., Cham, O.E.P., Galván, E.J.G.: A Review of Design Methodologies and Evaluation Techniques for FPGA-Based Visual Object Tracking Systems. International Journal of Combinatorial Optimization Problems and Informatics 15(5), 127–145 (2024). https://doi.org/10.61467/2007.1558.2024.v15i5.571
9. Smeulders, A.W., Chu, D.M., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.: Visual tracking: An experimental survey. IEEE Trans. Pattern Anal. Mach. Intell. 36(7), 1442–1468 (2013). https://doi.org/10.1109/TPAMI.2013.230
10. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys 38(4), 13 (2006). https://doi.org/10.1145/1177352.1177355
11. Christen, P., Hand, D.J., Kirielle, N.: A review of the F-measure: its history, properties, criticism, and alternatives. ACM Computing Surveys 56(3), 1–24 (2023)

12. Singh, U., Saini, A., Domala, R.: Object Tracking in Videos Using CNN. In: ICDSMLA 2019: Proc. of the 1st Int. Conf. on Data Science, Machine Learning and Applications, pp. 520–527. Springer, Singapore (2020)
13. Macek, K.: Pareto principle in datamining: an above-average fencing algorithm. Acta Polytechnica 48(6), 55–59 (2008)
14. Kumar, A.R., Ravindran, B., Raghunathan, A.: Pack and detect: Fast object detection in videos using region-of-interest packing. In: Proc. of the ACM India Joint Int. Conf. on Data Science and Management of Data, pp. 150–156 (2019)
15. An, E.B., Kim, A., Jung, S.H., Kwak, S., Lee, J.Y., Cheong, W.S., Seo, K.D.: Adaptive spatial down-sampling method based on object occupancy distribution for video coding for machines. EURASIP Journal on Image and Video Processing 36, 1–17 (2024)