Journal of Engineering Science and Technology
Vol. 17, No. 3 (2022) 1734 - 1747

STUDY AND INVESTIGATE THE CUCKOO OPTIMIZATION TO IMPROVE THE PARTICLE FILTER AS A BASIC TRANSITION MODEL ON SCALE VARIATION AND OCCLUSION OBJECT TRACKING

NAZRIA RAHMI, GRAFIKA JATI*, WISNU JATMIKO

Faculty of Computer Science, Universitas Indonesia,
Depok Campus Depok City, 16424, West Java, Indonesia
*Corresponding Author: grafikajati@ui.ac.id

Abstract

Scale variation and occlusion are still the main problems in visual object tracking. These problems arise because of uncertainty and the unpredictable movement of other objects around the target. A probabilistic-based tracker commonly deals with the problem. However, it is challenging to develop a robust transition model with a high-quality observation model. The proposed method is designing a cuckoo search to optimize Particle Filter as a base transition model. Cuckoo search spreading more various candidate target tracking based on Lévy Flights. The proposed method combined with affine transformation as particle representation and deep learning as an observation model. The proposed method achieves a precision of 0.894 and a success rate of 0.701 on the scale variation problem. It obtains a precision of 0.824 and a 0.621 success rate on occlusion, which is better than the baseline particle filters-based method. It also obtains competitive compared to the state-of-the-art method with seven times faster in computation. Robust Tracker in occlusion and scale variation becomes the fundamental base to real applications such as surveillance, robotics, and other intelligence systems.

Keywords: Cuckoo search, Deep learning, Object tracking, Occlusion, Particle filter, Scale variation.

1. Introduction

Surveillance systems, robot vision systems, self-driving cars, and other intelligent traffic systems apply object tracking technology [1, 2]. That capability estimates the target position in a video, or a sequence of images gathered from the camera. The surveillance system traditionally installs fix camera while the object tends to move. It causes the target scale various as objects to move closer to or away from the camera. This challenge is called scale variation, which changes a target size from large to small or vice versa [3]. Another problem occurs when the target realizes that it is being tracked. Target tries to avoid being tracked by moving behind the other object, so it is covered. This condition, known as occlusion, is when an object is covered by something, either entirely or partially closed [3]. Therefore, scale variation and occlusion are still a real challenge in the object tracking field [4].

Some researchers such as DFT(2012) [5], CSK(2012) [6], DLT(2013) [7], and ECO(2017) [8] deal with occlusion and scale variation problems by developing a robust in two aspects either transition model or observation model. The transition model defines a set of candidate targets based on object movement prediction among variables in space. At the same time, the observation model is a feature extraction stage to classify object targets.

Transition models are commonly allowed with search mechanisms consisting of two approaches: deterministic and probabilistic tracker [9]. Deterministic methods track objects for every frame using an iterative search to maximize the similarity between the region and the target window. One of the popular deterministic methods is namely the mean shift (MS) [10]. The probabilistic method assumes that the tracking process is a current state settlement. The popular method is the Bayesian modeling of the uncertainty and conditional density during the tracking process [11]. If the unpredictable motion of the target and object around it inflict occlusion and scale variation, the Bayesian Framework probabilistic method is more suitable for tracking.

The Bayesian framework has two numerical implementations, that is Kalman Filter and Particle Filter [11]. Kalman filter is an exact method that solves a problem by assuming the distribution is Gaussian. In contrast, Particle Filter utilizes an approximation approach, which assumes that the problem can have any distribution. However, target movement can be any distribution, so it is more suitable to use a particle filter approach [12].

Researchers utilize the optimization algorithm combined with Particle Filter to enhancing the search mechanism. Object motion has uncertainty, so the optimization algorithm enriches particle filter spreading particle on global optimal. Moreover, the optimization approach improves particle filter to solve the impoverishment problem. These problems arise because some low-weight particles are discarded while they may be potentially crucial from spatial distribution.

Higuchi [13] proposed a genetic algorithm (GA) where mutations and crossover operation in GA replaced the prediction steps. However, this algorithm performs is not significantly better than the sequential importance of resampling in particle filters. Park et al. [14] developed an evolutionary particles filter to solve sample impoverishment cases. Gao et al. [15] modified particle filters using a firefly algorithm (FA). It also successfully tracks the target in various challenging conditions like occlusion, complex background, and irregular motions, then

outperforms the conventional particle filter. However, this algorithm is difficult to tune due to this algorithm's parameters are susceptible to object tracking [16]. Zhang et al. [9] utilized Particle Swarm Optimization (PSO) to solve the problem of object tracking. The particles' movement control parameters in PSO are dynamically updated based on the particle fitness value. In current progress, Jati et al. [12] proposed Dynamic Swarm Particle, which optimizes Particle Filter using PSO to deal with fast motion but not good enough in occlusion.

Cuckoo Search (CS) is a nature-inspired optimization algorithm proposed by Yang and Deb [17]. CS mimics the Cuckoo parent parasite's lifestyle combined with the Lévy flight behavior pattern of several birds and fruit flies. Previous research reveals CS has better results compared to GA and PSO. CS algorithm has been used successfully in several optimization challenges such as structure design, grouping, economical shipping and prediction, and others [18]. CS had been applied in Particle Filter then obtain a better result than conventional particle filter [19]. Zhang et al. [9] compared Simulated Annealing (SA), Ant Lion Optimization (ALO), Particle Swarm Optimization (PSO), and Cuckoo Search (CS) in object tracking applications. They reported that CS achieves the highest precision compared to another optimization. Gui-Xia et al. [20] also improved particle filter using Cuckoo to solve the impoverishment problem. However, the two last optimized tracker uses a simple feature description using Histogram of Oriented Gradient (HOG). Therefore, it is hard to catch the target, especially in occlusion and scale variation.

Therefore, the tracker observation or appearance model also affects the tracking accuracy [7]. The occluded object only shows a small part of the object. At the same time, the scale variation makes the object's appearance is resized from frame to frame. The two challenges make it difficult to detect object targets both on the discriminative and generative tracker. Therefore, the tracker should construct a robust object descriptor to differentiate objects, although objects show partially or different scales.

Deep learning is currently prevalent and shows the capability of overcoming image classification [21] - the success of deep learning due to its deep architecture to learn the features of images. Deep Learning Tracker is a popular tracker that implements deep learning for feature descriptors [18], but it still uses the conventional Bayesian method [7]. This paper proposes an object tracker that modifies the transition model of particle filter using the CS approach combined with robust Stack Denoising Autoencoder for the observation model. The results are focused on dealing with the sequence that contains occlusion and scale variation challenges.

The paper is presented as follows. Section 2 describes the material and primary method of tracking and optimization algorithm, especially Cuckoo Search. In Section 3, the proposed method optimized deep particle filter using Cuckoo Search. Section 4 depicts the experiment result and discussion. Finally, the last Section 5 presents the conclusion.

2. Material and Methods

2.1. Object tracking framework

Object tracking is a process to predict the state variable x_t based on observations in time t. The estimation of state variable distribution obtained by the observation

value of the observation set $z_{1:t-1}$. In general, the algorithm of object tracking consists of input, transition models, observation models, and outputs which is shown in Fig. 1. [22].

- Input: The object target can be determined in the first frame. Another way the target object is obtained using object detection. In this research, the target object is given by determining the bounding box.
- Transition model: This model is a stage that collects a set of bounding box candidates containing the target object. It means that it represents target movements among frames or search space.
- Observation model: This model determines the best candidate based on similarity to the target object.
- Output: The result of tracking in the form of the target object's predicted bounding box.

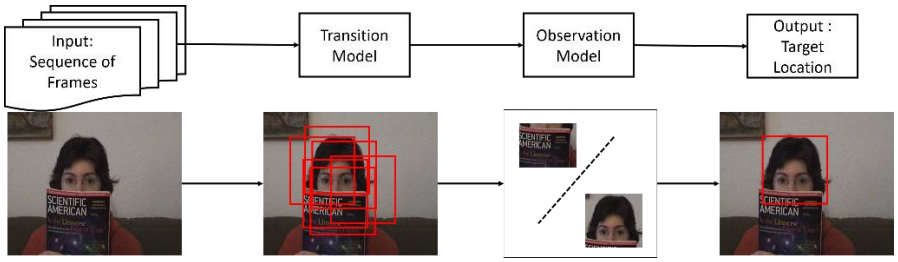


Fig. 1. The general framework of Bayesian-based object tracker.

2.2. Particle filter

The object tracking problem is outlined in the Bayes Theorem [11]. The distribution involved is assumed as Gaussian. This distribution can be solved by using the Kalman Filter method. Second, the numerical solution of the equation is known as Particle Filter [11]. The Particle filter (PF) is a Bayesian-based transition model that is widely applied in a tracker [12, 23-26]. Wang et al. [22] compared PF with Sliding windows and sliding window radius. PF obtain the best results compared with these two methods.

2.3. Cuckoo search

Particle Filter does iteration until finding the estimation of the target distribution. It consumes much time, so optimization accelerates the tracker to find target distribution at once find particle with the highest value that believes the target in tracking. The proposed method used the cuckoo search (CS) as optimization in the tracker motion model. The optimization algorithm with Cuckoo is enhanced by Lévy flights, where there are three rules in this algorithm [17, 27] namely:

i. Each Cuckoo lay a single egg at a time in a random nest
ii. The best nests with good egg quality affect the next generation
iii. The number of nests is fixed. There is a possibility that eggs placed by a cuckoo are known by the nest owner with a probability of p_a. Besides, there is a possibility that the egg is removed, or the nest owner leaves the nest and builds another nest.

Each egg and the nest are represented as a solution. Every egg from the Cuckoo that will be put in the nest is a new solution to replace a poor solution.

The new solution $x^{(t+1)}$ for Cuckoo, i is generalized using Lévy flight by Eqs. (1) and (2)

$$x_i^{t+1} = x_i^t + \alpha L(s, \lambda) \tag{1}$$

where

$$L(s, \lambda) = \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}} \tag{2}$$

where $\alpha > 0$ is the step size scaling factor associated with the scale of the problem. The α value indicates how big the steps to choose the nest that will be the solution. A small α value will select the prospective solution that is not far from the best solution to make the search process more effective. CS algorithm has a rule where there is a possibility that the nest owner will recognize the laid eggs, so the nest owner discards the eggs or leave and build a new nest. So the value for each nest or solution is updated with Eq. (3)

$$x_i^{t+1} = x_i^t + \alpha s \otimes H(p_a - \epsilon) \otimes (x_j^t - x_k^t) \tag{3}$$

where x_j^t and $x_k^t$are different solutions. $H\ (u)$ is the Hessian function, ϵ is a random value, and *s* is a step size.

3. Proposed Method

This paper proposes an optimized tracker, namely GDPSCuc. This tracker designs a geometry model for the transition model Particle Filter's input that employs deep learning as an observation model. It brings out GDPSCuc (Geometric Deep Particle Swarm Cuckoo) term because as a short-term tracker, GDPSCuc does not redetect the object when lost but uses Cuckoo Search to update particle location in the searching area. The target is manually defined in the first frame. Afterward, the tracker only depends on the first frame to extract the feature of the target.

3.1. Affine transformation as model representation

Object movement can be assumed to transform. Transformation occurs in two alternatives; the first is the transformation of objects in geometry, the second is the transformation of objects in the coordinate system. Object tracking objects represented objects are transforming in the geometry space. Affine is a transformation method suitable for tracking. This transformation is a linear combination of translation, scaling, shearing, and rotation operations. Whereas in object tracking, the shape of an object can change dynamically.

Affine transformation is a generalization of Euclidean Transformation. Affine transformation does not maintain the length and angle so that the shape of the object can change. For example, the line is transformed into a line in this transformation, while the circle can be transformed into an ellipse. Tracking objects with affine transformations on a particle filter is given as a form of geometry representation in a vector using local coordinates to make the particle filter easy to solve. The transformation that occurs in object tracking can be observed in Fig. 2.

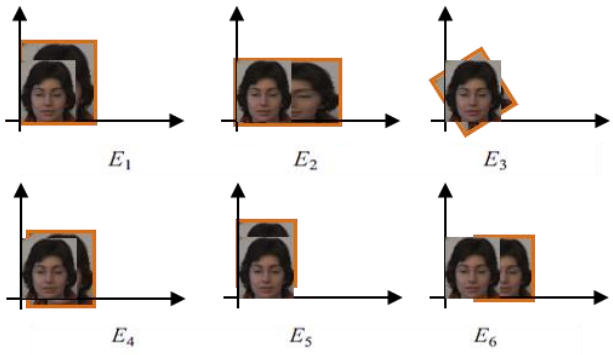


Fig. 2. Affine transformation.

3.2. Particle filter optimization with cuckoo search

The paper proposes an optimized particle filter transition model. The best particle search process will be carried out with cuckoo search (CS). However, initialization for CS engages with affine transformations. The dynamics used in this study are referred to as parameters, which include 6 parameters, namely: translation x, translation y, rotation, scaling, reflection, and shearing.

As a moving entity, particle motion can be limited by the speed of the object. This is done to help prevent particles from getting trapped at the local maximum. The speed of movement of the object can be obtained from first-order autoregressive (AR). AR is used in object tracking because it can approach the actual situation [28]. State equation with state dynamics can be written as the following Eqs. (4) and (5)

$$X_k = X_{k-1}.exp(A_{k-1}\Delta t + dW_k\sqrt{\Delta t}) \quad (4)$$

$$A_{k-1} = \beta \,\mathrm{Log}(X_{k-2}^{-1}.X_{k-1}) \quad (5)$$

where $\beta = 1$ is the AR parameter. The GDPSCuc main method is explained with the following algorithm modifications:

i. Particle initialization randomly around the last distribution generated by Particle Filter, so the particle is given an initial velocity based on Eq. (4).
ii. Each particle's fitness value is obtained using the observation process with a continuous value between minimal (0.0) and maximal (1.0). A value of 0.0 means that the object being the tracking target and the actual target have no resemblance. While the value of 1.0 indicates an accurate resemblance. In GDPSCuc, the method stops if the fitness value has reached 0.8.
iii. GDPSCuc update the position based on Eq. (1), where the value of i represents the value of the i solution $(i = 1,2,\dots,n)$, the value λ is a constant in this paper, the value $\lambda = 1.5$. While the value of s is calculated from the two Gaussian distributions U and V, calculated by the following Eq. (6) [27]

$$s = \frac{U}{|V|^{1/\lambda}} \quad (6)$$

where U and V are made randomly according to n many solutions. GDPSCuc defines n 50, which means that there are 50 particles represented as a solution in the CS algorithm. The value of α is 0f 0.01, which representation of how far the

steps of the solution. In this research, the target movement is assumed that the object displacement between frames is close.

iv. GDPSCuc discovers search space to find the best solution. The proposed tracker determines the probability p_a as a representation of the possibility of discovery, the smaller the value of p_a, the less likely for discovery to occur. Discovery uses Eq. (3). Based on the experiment, GDPSCuc set the value of p_a 0.25.

The proposed tracker GDPSCuc algorithmic process of optimizing CS on object tracking can be seen in Fig. 3.

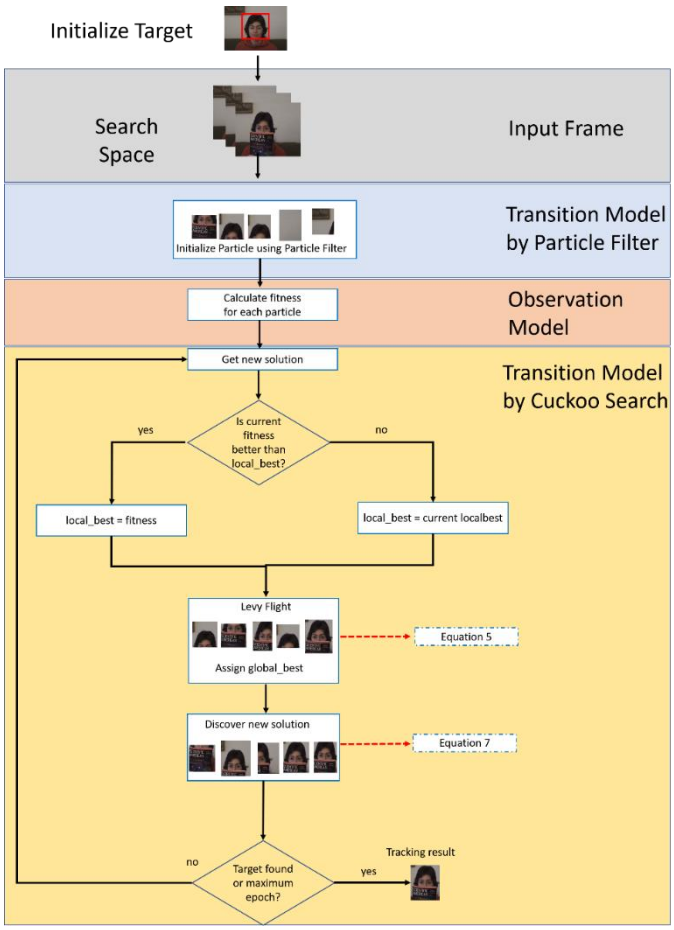


Fig. 3. Flowchart of Proposed method GDPSCuc.

3.3. Observation model using deep learning

The proposed tracker utilizes Stack Denoising Autoencoder (SDAE) to determine each particle's confidence [7]. Observation model help to get which one is the object. When the result is close to 1.0, it means that the candidate target is probably the real target defined in the first frame; if close to 0.0, it means the similarity is far from it. Our method combines particle filter as represented in parameter affine and optimize with the Cuckoo search that uses fitness as confidence value from SDAE.

3.4. Data and performance measurement

The proposed method is executed on a PC with specification Ubuntu 16.04 Operating System, Intel i5-7400, 3.0 GHz CPU with 24 GB RAM, and 8 GB NVIDIA GeForce GTX 1070 graphic card. The proposed method was implemented using Matlab software and tested using several sequences that contain scale variation and occlusion. Researcher test the method using several selected sequences in Table 1 from OTB50 and OTB100. Table 1. reports a sequence with scale variation, occlusion, or a combination of scale variation and occlusion problems. The sequence that contains a specific character is labelled with 1, then without it is labelled 0.

The proposed method is evaluated using the precision and success rate. Tracker result compared to the sequence ground truth. Precision calculates the difference between the center of the tracking result and the ground truth. Success rate calculates the percentage of bounding box overlap between ground truth and tracking results. The detain precision and success rate can be seen in Jati et al. [12].

Table 1. Sequence for testing tracker capability in scale variation and occlusion problem, taken from OTB50 and OTB100.

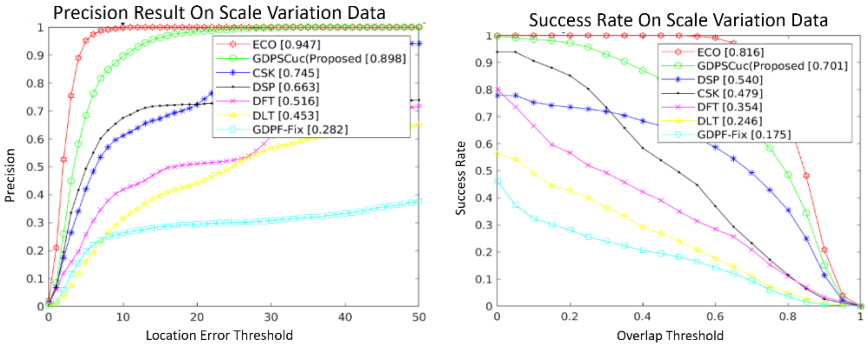
Sequence	Scale Variation (SV)	Occlusion (Occ)
Walking2	1	1
Walking	1	1
Freeman3	1	0
Dog1	1	0
Faceocc1	0	1
Football	0	1
Car4	1	0
Singer1	1	1
Car2	1	0
FaceOcc2	0	1

4. Result and Discussion

The experiment is conducted to show GDPSCuc performance dealing with scale variation and occlusion. GDPSCuc is compared with baseline and other notable trackers. The Deep Learning Tracker (DLT) is the baseline method that uses a conventional particle filter as a transition model and deep learning as the observation [7]. Gunawan et al. [29] enhanced DLT transition using movement velocity target become new tracker, namely Geometric Deep Particle Filter-Fix (GDPF-Fix). The current modification in DLT is Dynamic Swarm Particle (DSP) that optimizes Particle Filter using Particle Swarm Optimization. For notable trackers selected from the tracker with an impressive performance in precision and speed, such as ECO [8], DFT [5], and CSK [6].

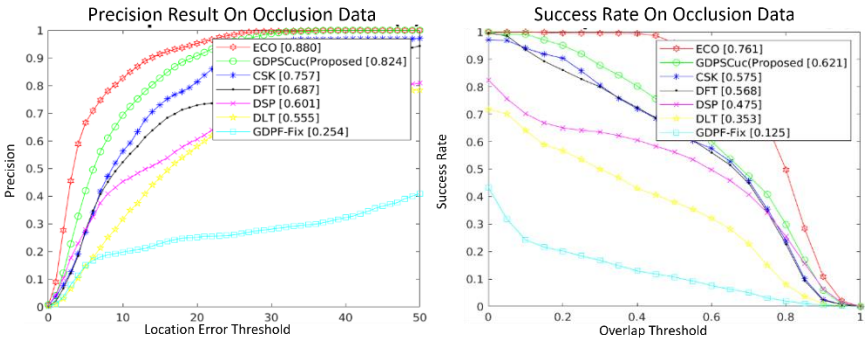
4.1. Baseline tracker comparison

It is comparing to the base method that is DLT, GDPF-Fix, and DSP. The proposed method GDPSCuc obtain the best result. Fig. 4 shows that GDPSCuc (proposed tracker) better than the base method in scale variation that is 0.898 and 0.701 for precision and success rate. In occlusion, the problem is shown in Fig 5. GDPSCuc also gets higher precision and success rate that is 0.824 and 0.621, respectively.



(a) **(b)**

Fig. 4. Tracking result of GDPSCuc and state-of-the-art tracker on scale variation problem (a) precision (b) success rate.



(a) **(b)**

Fig. 5. Tracking result of GDPSCuc and state-of-the-art tracker on occlusion problem (a) precision (b) success rate.

GDPSCuc can find the target although object size change over the sequence. GDPCCus has a discovery step that generates new particles with various objects' sizes based on affine transformations. DSP also utilizes affine transformations. However, DSP, as the latest Particle optimization, uses PSO optimization. Cuckoo Search optimization using Levy flight to balance exploitation and exploration of candidate target. It is better than PSO, which uses a speed of particle to explore another solution.

On the occlusion problem, the GDPSCuc observes each Cuckoo particle using a deep learning approach. The observation model is an essential part of determining the best particle and stopping tracker iteration. In Fig. 5, the GDPSCuc is still better in terms of precision and success rate. So, GDPSCuc successfully enhances performance between baseline methods such as DLT, GDPF-Fix, and DSP.

Table 2 shows that DLT does not take a long computational time because DLT only uses basic particle filters, so it does not have many iteration processes. Then underneath, there is GDPF-Fix, which is an enhancement of DLT, the fps of this method is reduced because this method adds a bootstrap particle filter and affine

transformation. After that, there is DSP as an advanced method of GDPF-Fix. The DSP is given additional optimization so that this slows down computing time. The GDPSCuc takes more time than DSP because the optimization process between PSO on DSP and CS on GDPFCuc is principally different. CS searches for solutions twice in one iteration, while PSO only once.

Table 2. Recap precision, success rate, and computation speed in frame per second (*fps*) of GDPSCuc and *other* trackers.

No.	Method	Precision		Success Rate		fps
		SV	Occ	SV	Occ	
1	ECO	0.947	0.880	0.816	0.761	2,02
2	GDPSCuc	0.898	0.824	0.701	0.621	14,58
3	CSK	0.745	0.757	0.479	0.575	264,54
4	DSP	0.663	0.601	0.540	0.475	17,95
5	DFT	0.516	0.687	0.354	0.568	23,51
6	DLT	0.453	0.555	0.246	0.353	25,42
7	GDPF_Fix	0.282	0.254	0.175	0.125	18,47

4.2. State-of-the-art tracker comparison

GDPSCuc is also compared with other methods that have a different principle. Fig. 4 and Fig. 5 show that GDPSCuc reaches the second-best right after ECO in the first position. ECO obtain 0.947 for precision and 0.816 for success rate in scale variation problem. The best position is also taken by ECO in occlusion with 0.880 for precision and 0.761 for success rate. GDPSCuc still competitively in precision and success rate evaluation.

Moreover, Table 2 depicts that GDPSCuc runs 14.58 frames per second while ECO runs 2.02 frames per second, so GDPSCuc is 7 times faster than ECO in terms of computation speed. GDPSCuc reaches processing speed for real-time application, which is around 15-30 fps [30]. Therefore, GDPSCuc can be implemented in real-time applications such as surveillance systems and advanced driver assistance systems (ADAS).

ECO is developed based on a Correlation Filter that has more time complexity than Particle Filter based. Although each particle in GDPSCuc is evaluated using Stack Denoising Auto Encoder, GDPSCuc can reduce the resampling step with the Cuckoo search's discovery step. Therefore, it significantly reduces computation speed.

Table 2 also shows that CSK is the fastest tracker because it uses Fast Fourier Transform as a tracking-based detection method. However, CSK obtains precision and success rate value under GDPSCuc. Especially in success rate evaluation, this condition due to CSK cannot adjust the bounding box size. Finally, GDPSCuc tracking on scale variation and occlusion is shown in Fig 6. Fig. 6(a) shows that the bounding box tracker is adjusted following object size. Although the object moves closer or away from the camera, GDPSCuc can follow the target size. In another condition, Fig. 6(b) shows that GDPSCuc is successfully handling occlusion. It can be seen that several occlusion scenarios from the sequence, namely faceOcc1. Then, Fig. 6(c) shows that the proposed method can track an object with a scale variation problem.

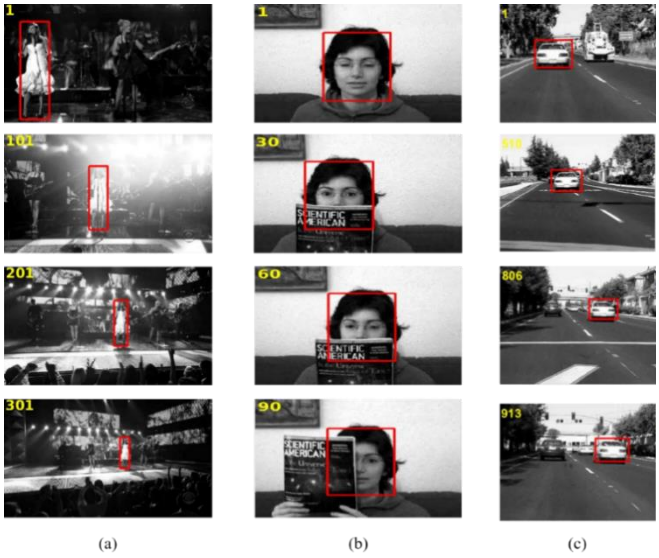


Fig. 6. GDPSCuc tracking result: (a) singer1(scale variation and occlusion), (b) faceOcc1(occlusion), (c) Car2 (scale variation).

5. Conclusion

The proposed method, GDPSCuc, successfully enhances the performance Particle-Based transition model using Cuckoo Search. GDPSCuc also utilized the deep learning method as an observation model that robust selected the best particle of Cuckoo search as a tracking result. Based on the experiment, GDPSCuc achieves precision above 0.80 for scale variation and occlusion. GDPSCuc also obtains 0.70 and 0.621 both in success rate. This result is higher than the baseline method. Compared to one of the notable state-of-the-art methods, namely ECO, GDPSCuc is slightly below precision and success rate, but GDPSCuc excels in computation speed 7 times faster than ECO.

Acknowledgments

This work was supported in part by PUTI Q3 Universitas Indonesia 2020 Grant entitled " Pedestrian Tracking using Geometric Deep Swarm Cuckoo for Self-Driving Car " with number: NKB-4378/UN2.RST/HKP.05.00/2020.

Nomenclatures

A_{k-1}	Particle velocity in time k-1
dW_k	The random error of particle position
dx_{t-1}	Derivation with respect to x_{t-1}
H	Hessian function
$P()$	Probability
p_a	Possibility of discovery in Cuckoo search
s	Step size in Cuckoo Search
t	time

U	Gaussian distribution named U
V	Gaussian distribution named V
$x_{1:t}$	State from 1 to t
$x_{1:t-1}$	State from 1 to t-1
X_k	State or particle position in time k, which is dynamic
X_{k-1}	State or particle position in time k-1, which is dynamic
X_{k-2}^{-1}	State transformation
x_t	State in t
x_i^t	Solution of Cuckoo i in t
x_j^t	Solution of Cuckoo j in t
x_k^t	Solution of Cuckoo k in t
$x^{(t+1)}$	The cuckoo solution in t+1
$z_{1:t}$	Observation from 1 to t
x_i^{t+1}	Solution of Cuckoo i in t+1
$z_{1:t-1}$	Observation from 1 to t-1
z_t	Observation in t

Greek Symbols

α	Coefficient of step size in the Cuckoo search
β	Autoregressive Parameter to show how much velocity involved in particle movement
λ	Gaussian distribution constant in Cuckoo Search
ϵ	Random value

Abbreviations

ALO	Ant Lion Optimization
AR	Autoregressive
CS	Cuckoo Search
CSK	Circulant Structure with Kernels
DFT	Distribution fields Tracking
DLT	Deep Learning Tracker
DSP	Dynamic Swarm Particle
ECO	Efficient Convolution Operators
FA	Firefly Algorithm
GA	Genetic Algorithm
GDPF-Fix	Geometric Deep Particle Filter-Fix
GDPSCuc	Geometric Deep Particle Swarm Cuckoo
HOG	Histogram of Oriented Gradient
PF	Particle Filter
PSO	Particle Swarm Optimization
SA	Simulated Annealing
SDAE	Stack Denoising Auto Encoder

References

1. Velastine, S.A.; Fernández, R.; Espinosa, J.E.; and Bay, A. (2020). Detecting, tracking and counting people getting on/off a metropolitan train using a standard video camera. *Sensor*, 20(21), 6251.

2. Amirullah, A.; Bakti, R.U.; and Areni, I.S. (2018). A modified pinhole camera model for automatic speed detection of diagonally moving vehicle histogram of oriented gradient. *Journal of Engineering Science and Technology* (*JESTEC*), 13(6), 1722-1734.
3. Wu, Y.; Lim, J.; and Yang, M.-H. (2015). Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9), 1834-1848.
4. Kristan, M. et al. (2020). The eighth visual object tracking VOT2020 challenge results. *European Conference on Computer Vision*. Springer, Cham. https://doi.org/10.1007/978-3-030-68238-5_39
5. Sevilla-Lara, L.; and Learned-Miller, E. (2012). Distribution fields for tracking. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Providence, RI, USA, 1910-1917.
6. Henriques, J.F.; Caseiro, R.; Martins, P.; and Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. *Lecture Notes in Computer Science* (*including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*); 7575, part 4, 702-715.
7. Wang, N.; and Yeung, D.-Y. (2013). Learning a deep compact image representation for visual tracking. *Advances in Neural Information Processing Systems*, 26, 1-9.
8. Danelljan, M.; Bhat, G.; Khan, F.S.; and Felsberg, M. (2017). ECO : Efficient convolution operators for tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, 6638-6646.
9. Zhang, H.; Zhang, X.; Wang, Y.; Shi, K.; Zhang, J.; and Li, C. (2018). An experimental comparison of swarm optimization based abrupt motion tracking methods. *IEEE Access*, 6, 75383-75394.
10. Sun, J. (2012). A fast MEANSHIFT algorithm-based target tracking system. *Sensor*, 12(3), 8218-8235.
11. Haug, A.J. (2012). *Bayesian estimation and tracking: A practical guide*. Wiley.
12. Jati, G.; Gunawan, A.A.S.; and Jatmiko, W. (2019). Dynamic swarm particle for fast motion vehicle tracking. *ETRI Journal*, 42(1), 54-66.
13. Higuchi, T. (2007). Monte Carlo filter using the genetic algorithm operators. *Journal of Statistical Computation and Simulation*, 59(1), 37-41.
14. Park, S.; Hwang, J.P.; Kim, E.; and Kang, H.-J. (2009). A new evolutionary particle filter for the prevention of sample impoverishment. *IEEE Transactions on Evolutionary Computation*, 13(4), 801-809.
15. Gao, M.-L.; Li, L.-L.; Sun, X.-M.; Yin, L.-J.; Li, H.T.; and Luo, D.S. (2015). Firefly algorithm (FA) based particle filter method for visual tracking. *Optik*, 126(18), 1705-1711.
16. Gao, M.; He, X.; Luo, D.-S.; Jiang, J.; and Teng, Q.-Z. (2013). Object tracking using firefly algorithm. *IET Computer Vision*, 7(4), 227-237.
17. Yang, X.-S.; and Deb, S. (2009). Cuckoo search via Lévy flights. 2009 *World Congress on Nature & Biologically Inspired Computing* (*NaBIC*), Coimbatore, India, 210-214.
18. Piechocki, J.; Ambroziak, D.; Palkowski, A.; andRedlarski, G. (2014). Use of modified Cuckoo search algorithm in the design process of integrated power

systems for modern and energy self-sufficient farms. *Applied Energy*, 114(C), 901-908.

19. Gao, M.-L.; Yin, L.-J.; Zou, G.-F.; and Liu, W. (2015). Visual tracking method based on cuckoo search algorithm. *Optical Engineering*, 54(7), 073105.
20. Gui-Xia, F.; Ming-Liang, G.; Guo-Feng, Z.; Wen-Can, L.; and Li-Na, L. (2018). An improved particle filter based on cuckoo search for visual tracking. 2018 *Chinese Control and Decision Conference* (*CCDC*), Shenyang, China, 3687-3691.
21. He, K.; Zhang, X.; Ren, S.; and Sun, J. (2016). Deep residual learning for image recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 770-778.
22. Wang, N.; Shi, J.; Yeung, D.-Y.; and Jia, J. (2015). Understanding and diagnosing visual tracking systems. *IEEE International Conference on Computer Vision Understanding* (*ICCV*), Santiago, Chile, 3101-3109.
23. Jati, G.; Gunawan, A.A.S.; Jatmiko, W.; and Febrian, A. (2016). Accurate visual tracking by combining Bayesian and evolutionary optimization framework. *International Conference on Advanced Computer Science and Information Systems*, Malang, Indonesia, 523-528.
24. Gunawan; A.A.S.; Jatmiko, W.; and Arymurthy, A.M. (2017). Fast and optimal visual tracking based on spectral method. *Procedia Computer Science*, 116, 571-578.
25. Lahraichi, M.; Housni, K.; and Mbarki, S. (2018). Visual tracking using particle filter based on Gabor features. *International Journal of Intelligent Engineering and Systems*, 11(4), 147-157.
26. Xu, Y.; Zhou, X.; Chen, S.; Li, F. (2019). Deep learning for multiple object tracking: a survey. *IET Computer Vision Review*, 13(4), 355-368.
27. Yang, X.-S. (2014). *Nature-inspired optimization algorithms*. London, United Kingdom, Elsevier.
28. Reynard, D.; Wildenberg, A.; Blake, A.; and Marchant, J. (1996). Learning dynamics of complex motions from image sequences. *Lecture Notes in Computer Science* (*including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*), 357-368.
29. Gunawan; A.A.S. and Jatmiko, W. (2015). Geometric deep particle filter for motorcycle tracking: Development of intelligent traffic system in Jakarta. *International Journal on Smart Sensing and Intelligent Systems*, 8(1), 429-463.
30. Velez, G.; and Otaegui, O. (2016). Embedding vision-based advanced driver assistance systems : A survey. *IET Intelligent Transport Systems*, 11(3), 103-112.