Information Sciences 283 (2014) 267–287

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Refined particle swarm intelligence method for abrupt motion tracking

Mei Kuan Lim [a,*], Chee Seng Chan [a], Dorothy Monekosso [b], Paolo Remagnino [b]

[a] Center of Image and Signal Processing, University of Malaya, 50603 Kuala Lumpur, Malaysia
[b] Computing and Information Systems, Kingston University, Surrey KT1 2EE, United Kingdom

ARTICLE INFO

Article history:
Received 10 July 2013
Received in revised form 17 December 2013
Accepted 3 January 2014
Available online 5 April 2014

Keywords:
Abrupt motion tracking
Visual tracking
Particle swarm optimisation
Computer vision

ABSTRACT

Conventional tracking solutions are not able to deal with abrupt motion as these are based on a smooth motion assumption or an accurate motion model. Abrupt motion is not subject to motion continuity and smoothness. We address this problem by casting tracking as an optimisation problem and propose a novel abrupt motion tracker based on swarm intelligence – the SwATrack. Unlike existing swarm-based filtering methods, we first of all introduce an optimised swarm-based sampling strategy for a tradeoff between the exploration and exploitation of the state space in search for the optimal proposal distribution. Secondly, we propose Dynamic Acceleration Parameters (DAP) that allow on the fly tuning of the best mean and variance of the distribution for sampling. Combining the two strategies within the Particle Swarm Optimisation framework represents a novel method to address abrupt motion. To the best of our knowledge, this has never been done before. Thirdly, we introduce a new dataset – the Malaya Abrupt Motion *(MAMo)* dataset that consists of 12 videos with groundtruth. Finally, experimental on both quantitative and qualitative results have shown the effectiveness of the proposed method in terms of dataset unbiased, object size invariant and fast recovery in tracking the abrupt motions.

1. Introduction

Visual tracking is one of the most important and challenging research topics in computer vision. Its importance stems from the fact that it is pertinent to the tasks of motion based recognition, automated surveillance, video indexing, human–computer interaction and vehicle navigation [36,37]. In general, motion estimation in a typical visual tracking system can be formulated as a dynamic state estimation problem: $x_t = f(x_t - 1, v_t - 1)$ and $z_t = h(x_t, w_t)$, where x_t is the current state, f is the state evolution function, v_t is the evolution process noise, z_t is the current observation, h denotes the measurement function, and w_t is the measurement noise. The task of motion estimation is usually implemented by utilising predictors such as kalman filters [32,29,22], particle filters [11,2,18,3,4], or linear regression techniques [7]. These predictors are commonly enhanced by assuming that motion is always governed by a Gaussian distribution based on Brownian motion or constant velocity motion models [37,10].

While this assumption holds true to a certain degree for smooth motion, it tends to fail in the case of abrupt motion such as inconsistent speed (e.g. the movement of ball in sport events), camera switching (tracking of subject in a camera topology)

* Corresponding author. Tel./fax: +60 379676433.
E-mail addresses: imeikuan@siswa.um.edu.my, imeikuan@yahoo.com (M.K. Lim).

http://dx.doi.org/10.1016/j.ins.2014.01.003
0020-0255/

and low frame-rate videos, as illustrated in Fig. 1. The main reason is that the state equation cannot cope with the unexpected dynamic movement, e.g. sudden or sharp changes of the camera/object motion in adjacent frames. These sampling-based solutions also suffer from the well-known local trap problem and particle degeneracy problem. In order to handle these problems, one of the earliest work [16] considered tracking in low frame rate videos. Their work considers tracking in low frame rate as abrupt motion, and proposed a cascade particle filters to solve this problem. This is then followed by a number of sampling strategies [12–14,40,41,34,30], which are incorporated into the standard Markov Chain Monte Carlo (MCMC) tracking framework. Their method alleviates the constant velocity motion constraint in MCMC by improvising the sampling efficiency.

The aforementioned works have shown satisfactory results in tracking abrupt motion. However, we observed that most of the work involved applying different sampling strategies to the Bayesian filtering framework. There is a clear trend towards increase complexity; as methods become more complicated to cope with more difficult tracking scenarios. Often these sophisticated methods compensate the increased in complexity by trading-off performance in some other area. For example, the increased number of subregions for sampling to cope with the variation of abrupt motion is compensated by using a smaller number of samples to reduce, if not maintaining, the computational cost incurred. *However, are these complex and sophisticated methods really necessary?*

Recently, Particle Swarm Optimisation (PSO) [6,28,39,26,21,25], a new population based stochastic optimisation technique, has received more and more attention because of its considerable success. Unlike the independent particles in the particle filter, the particles in PSO interact locally with one another and with their environment by using the analogy of the cooperative aspect of social behaviours of animal swarm. For example, the flocking and schooling patterns of birds and fish. However, the standard PSO is not able to track abrupt motion efficiently, due to swarm explosion and divergence problems when the motion is highly abrupt [15,38,17].

In this paper, we proposed the SwATrack – Swarm intelligence-based Tracking algorithm to handle the abrupt motion. Our contributions are firstly, in contrast to the conventional solutions that are based on the different sampling methods in Bayesian filtering which are computationally expensive, we cast the problem of tracking as an optimisation problem and adopted the particle swarm optimisation algorithm as the sole motion estimator. In particular, we replace the state equation, $x_t = f(x_t - 1, v_t - 1)$ with a novel velocity model which is estimated by the PSO. Secondly, we introduced Dynamic Acceleration Parameters (DAP) and Exploration Factor ($\mathcal{E}$) into the proposed PSO framework to avoid the swarm explosion and divergence problems when tracking highly abrupt motion. While the standard PSO algorithm is not new, the novelty is in combining the DAP and $\mathcal{E}$ in an ingenious way to handle the abrupt motion, which is worth noting. To the best of the authors knowledge, there has yet to be published work with similar idea. Thirdly, a new abrupt motion dataset namely as the Malaya Abrupt Motion *(MAMo)* dataset with ground truth is introduced. The dataset consists of 12 videos in real and synthetic environment. Finally, experimental results and comparison with the state-of-the-art algorithms have shown the effectiveness and robustness of the proposed method in terms of dataset unbiased, object size invariance and recovery from error.

Fig. 1. Example of the abrupt motion in different scenarios. Top: Inconsistent speed. Middle: Camera switching. Bottom: Low frame-rate videos.

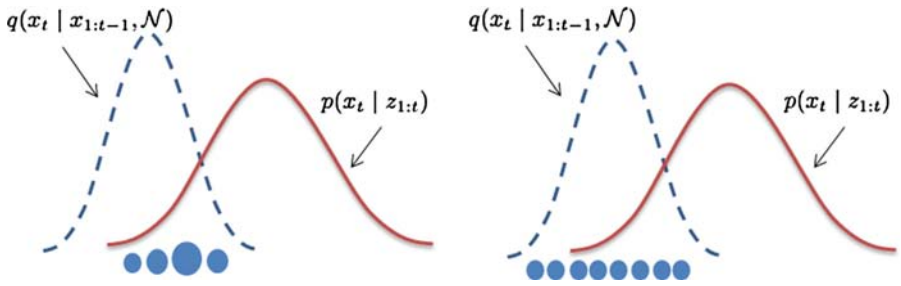
The rest of this paper is organised as follows. In Section 2, we provide the background work in tracking abrupt motion. The standard PSO is revisited in Section 3 and its limitations in handling abrupt motion is discussed. The proposed work is detailed in Section 4 while experimental results and discussion are given in Sections 5 and 6, respectively. Finally, the conclusion is drawn in Section 7.

2. Related work

While considerable research exist in relation to visual tracking, only a handful deal with abrupt motion [33,14,19,41]. Abrupt motion can be defined as situations where the object motion changes between adjacent frames with unknown patterns in scenarios such as (i) partially low-frame rate, (ii) switching of camera views in a topology network or (iii) irregular motion of the object. Therefore, conventional sampling-based solutions that assume Gaussian distribution based on Brownian motion or constant velocity motion models tend to fail in this area as illustrated in Fig. 2. In this paper, the literature review section is focused on works that handle abrupt motion only. Thus, for a complete review on general visual tracking, we encourage readers to refer to [36,37].

In recent work, Markov Chain Monte Carlo (MCMC) was used to overcome the computational complexity in PF as the state space increases [43]. While MCMC methods cope better in a high-dimensional state space, a common problem is the need to have a large number of samples, especially when tracking abrupt motion. Thus, to deal with abrupt motion, there a number of researchers who introduced modifications and refinements to the conventional MCMC. Kwon et al. in [12], integrate the Wang-Landau algorithm into the MCMC tracking framework to track abrupt motion. Their method alleviates the constant-velocity motion constraint in MCMC by improvising the sampling efficiency using the proposed annealed Wang-Landau Monte Carlo (A-WLMC) sampling method. The A-WLMC method increases the flexibility of the proposal density in MCMC by utilising the likelihood and density of states terms for resampling. Yet, another variation of MCMC known as the interactive MCMC (IMCMC) was proposed [13], where multiple basic trackers are deployed to track the motion changes of a corresponding object. The basic trackers which comprise of different combinations of observation and motion models are then fused into a compound tracker using the IMCMC framework. The exchange of information between these trackers has been shown to cope with abrupt motion while retaining the number of samples used. In another advancement, a highly adaptive MCMC (IA-MCMC) sampler [41] has been proposed. Their method further reduces the number of samples required when tracking abrupt motion by performing a two-step sampling scheme; the preliminary sampling step discovers the rough landscape of the proposal distribution (common when there is large motion uncertainty in abrupt motion) and the adaptive sampling step refines the sampling space towards the promising regions found by the preliminary sampling step. In another attempt for effective sampling of abrupt motion, [14] proposed the N-fold Wang-Landau (NFWL) tracking method that uses the N-fold algorithm to estimate the density of states which will then be used to automatically increase or decrease the variance of the proposal distribution. The NFWL tracking method copes with abrupt changes in both position and scale by dividing the state space into larger number of subregions. The N-fold algorithm was introduced during sampling to cope with the exponentially increased number of subregions.

Motivated by the meta-level question prompted in [42] on *whether there is a need to have more training data or better models for object detection*, we raise similar question in this domain; will continued progress in visual tracking be driven by the increased complexity of tracking algorithms? As indicated in the earlier section, often these sophisticated methods compensate the increased in complexity in a certain aspect of the algorithm by reducing another aspect of it. Furthermore, according to [10], different scenarios require different dynamic models. If *motion models only work sometimes*, on a particular scenario, then how far should the increased in complexity of tracking algorithms be, in order to cope with the challenges of real-time tracking scenarios? Should we look into less complex methods instead, since motion models only work sometimes? Hence, we study a simple and yet effective algorithm, the SwATrack that utilise the PSO framework to effectively handle the abrupt motion using the particles sharing information themselves. We cast tracking as an optimisation problem, and hence the proposed method is dataset unbiased, invariance to object size and able to recover from error.



(a) Particle degeneracy (b) Trapped in local optima

Fig. 2. Known problem of sampling-based tracking such as particle filter tracking and its variation.

Work that are considered similar to us are [15,38]. Li et al. [15] proposed a two-layers tracking framework in which PSO is successfully combined with a level set evolution. In the first layer, PSO is used to capture the global motion of the target and to help construct the coarse contour. In the second layer, level set evolution based on the coarse contour is carried out to track the local deformation. However, there is a possibility that some samples will get trapped in a few strong local maxima. Hence, the PSO method fails to track highly abrupt motions. Zhang et al. [38] proposed a swarm intelligence based particle filter algorithm with a hierarchical importance sampling process which is guided by the swarm intelligence extracted from the particle configuration, and thus greatly overcome the sample impoverishment problem suffered by particle filters. Nevertheless, their method is not ideal as it is still dependent on the Gaussian approximation. In order to handle this issue, we introduce (1) DAP – a dynamic acceleration parameter utilising the averaged velocity information of the particles; and (2) $\mathcal{E}$ – a mechanism for introducing a tradeoff between exploration and exploitation of the swarm into the PSO framework. With this, the SwATrack is to minimise the effect of the local trapped and sample impoverishment problems, while at the same time, is able to track highly abrupt motion and recover from tracking error.

3. Particle swarm optimisation revisit

Particle Swarm Optimisation (PSO) – a population-based stochastic optimisation technique was developed by Kennedy and Eberhart in 1995 [6]. It was inspired by the social behaviour of a flock of birds. Briefly, let us assume a n-dimensional search space, $\mathcal{S} \subset \mathcal{R}^n$ and a swarm comprising of I particles. Each particle represents a candidate solution to the search problem and is associated to a fitness function (cost function), $f : \mathcal{S} \rightarrow \mathcal{R}$. At every kth iteration, each particle is represented as $\{x_k^i\}_{i=1,\ldots,I}$, where $k = 1, 2, \ldots K$. Each particle, x_k^i has its own velocity, $v(x_k^i)$ and a corresponding fitness value (cost), $f(x_k^i)$. The best position encountered by the ith particle (personal best) will be denoted as $\{p(x_k^i)\}_{i=1,\ldots,I}$ and the fitness value as $pBest_k^i = f(p(x_k^i))$. For every kth iteration, the particle with the best fitness value will be chosen as the global best and is denoted as the index of the particle, which is denoted as f. Finally, the overall best position found by the swarm is represented as $gBest_k^i = f(p(x_k^g))$. The PSO algorithm is shown in Algorithm 1.

Algorithm 1. Standard PSO

Initialisation, at iteration k = 0
- Initialise a population of I particles, $\{x_k^i\}_{i=1,\ldots,I}$ with positions, $p(x_k^i)$, at random within the search space, $\mathcal{S}$.
- Initialise the velocities, $v(x_k^i)$ at random within $[1, -1]$.
- Evaluate the fitness value of each particle and identify their personal best $pBest_k^i = f(p(x_k^i))$.
- Identify the global best gth particle and update the global best information, $gBest_k = f(p(x_k^g))$.

for $k = 1$ to K **do**
  for $i = 1$ to I **do**
    Compute the new velocity according to:

$$v_{k+1}^i = \left[(\omega * v_k^i) + \left(c_1 * r_1 * \left(pBest_k^i - x_k^i\right)\right) + (c_2 * r_2 * (gBest_k - x_k^i)\right] \tag{1}$$

    Update the position according to:

$$p(x_{k+1}^i) = p(x_k)^i + v_{k+1}^i \tag{2}$$

    Check for out of bound:

$$p(x_{k+1}^i) \subset \mathcal{S} \tag{3}$$

    Update variables, $pBest_k^i, p\left(pBest_k^i\right), g, gBest_k, p(gBest_k^g)$
    Check for **Convergence**
    if Convergence == TRUE **then**
      Terminate iteration
    else
      Continue
    end if
  end for
end for

In Eq. (1), the parameters ω, c_1 and c_2 are positive acceleration constants used to scale the influence of the inertia, cognitive and social components respectively; $r_1, r_2 \subset (0, 1)$ are uniformly distributed random numbers to randomise the search exploration.

3.1. Limitations of PSO in tracking abrupt motion

As aforementioned, the traditional PSO is not able to cope with abrupt motion, due to:

Constant acceleration parameters: The parameter c_1 controls the influence of the cognitive component, $\left(c_1 * r_1 * \left(pBest_k^i - x_k^i\right)\right)$ which represents the individual memory of particles (personal best solution). A higher value of this parameter indicates a bias towards the cognitive component and vice versa. On the other hand, the parameter c_2 controls the influence of the social component, $(c_2 * r_2 * (gBest_k - x_k^i)$ which indicates the joint effort of all particles to optimise a particular fitness function, f.
The main drawback of the current PSO is the lack of a reasonable mechanism to effectively handle the acceleration parameters (ω, c and r); which are often set to constant variables [5,8,31]. For example, many applications of the PSO and its variant set these values to, $c_1 = c_2 = 2.0$, which gives the stochastic factor a mean of 1.0 and giving equal importance to both the cognitive and social components [6]. This limits the search space and therefore cannot cope with abrupt motion. Therefore, it is essential to have dynamic acceleration parameters that are able to cope better with the unexpected dynamics in abrupt motion.
Tradeoffs in exploration and exploitation: The inertia weight, ω plays an important role, directing the exploratory behaviour of the swarms. A high value of inertia accentuates the influence of the previous velocity information and forces the swarm to explore a wider search space; while a decreasing inertia weight reduces the influence of the previous velocity and exploit a smaller search space. Often, the inertia value that controls the influence of the previous velocity is set to $\omega \in [0.8, 1.2]$ [23]. Recently, decaying inertia weight, $\omega = 0.9 \rightarrow 0.4$ have been proposed and tested, with the aim of favouring global search at the start of the algorithm and local search later. While these settings have been shown to work well in other optimisation problems, one must note that it is not applicable to tracking abrupt motion where the dynamic change is unknown. Therefore, a solution that is able to handle the tradeoffs between the exploration and exploitation is crucial.

4. Proposed method – SwATrack

In this section, we present our proposed SwATrack – a variant of the traditional PSO to track target with arbitrary motion. Particularly, we will discuss how the effective combination of Dynamic Acceleration Parameters (DAP) and Exploration Factor $\mathcal{E}$ in the proposed PSO framework can alleviate the problem of swarm explosion and divergence problem.

4.1. Dynamic Acceleration Parameters (DAP)

PSO is a population based stochastic technique. Since PSO is an iterative solution, efficient convergence is an important issue towards a real-time abrupt motion estimation system. However, the strict threshold of the conventional PSO velocity computation as in Eq. (4) will always lead to particles converging to a common state estimate (the global solution). One reason is that the velocity update equation uses a decreasing inertia value which indirectly forces the exploration of particles to decrease over the iterations. On the other hand, an increasing inertia value will lead to swarm explosion in some scenarios.

To overcome this, we introduce DAP – a mechanism to self-tune the acceleration parameters by utilising the averaged velocity information of the particles. We normalise the acceleration parameters so that they can be compared fairly with respect to the estimated velocity, $p(w \cap c_1 \cap c_2) = 1.0$. The fitness function information is incorporated in the PSO framework in order to refine the acceleration parameters dynamically, according to the quality of estimation, rather than employing a static value. The basic idea is that when an object moves consistently in a particular direction, $\mathcal{C} \rightarrow 1.0$, the inertia, w and cognitive weight, c_1 values are increased to allow resistance to any changes in its state of motion in the later frames. Otherwise when $\mathcal{C} \rightarrow 0$, the social weight c_2 is increased by a step size to reduce its resistance to motion changes as Eq. (4). The increase of the social weight allows global influence and exploration of the search space, which is relevant when the motion of a target is dynamic. The exploitation within nearby regions is equitable when an object is moving with consistent motion.

$$\begin{cases} c_1 = c_1 + m;\ c_2 = c_2 - m;\ \omega = \omega + m & C \rightarrow 1.0 \\ c_1 = c_1 - m;\ c_2 = c_2 + m;\ \omega = \omega - m & otherwise \\ {}^{*}\text{subject to } p(\omega \cap c_1 \cap c_2) = 1.0 \end{cases} \tag{4}$$

The C is estimated by computing the frequency of the change in the quantised motion direction of the object; $\mathcal{C} \rightarrow 1.0$ represents consistent motion with minimal change of direction, while $\mathcal{C} \rightarrow 0$ represents inconsistent or dynamic motion.

4.1.1. Exploration factor ($\mathcal{E}$)

The normalisation of DAP to 1.0 will restrict the overall exploration of the state to a certain degree. Hence, we refine this by introducing the exploration factor, $\mathcal{E}$ which serves as a multiplying factor to increase or decrease the exploration. We define the exploration factor, $\mathcal{E}$ as the parameters that adaptively.

1. increase the *exploration* with high variance, and
2. increase the *exploitation* with low variance.

By utilising these exploitation and exploration capabilities, our method is capable of recovering from being trapped in a common state (local optima). Thus, the proposed SwATrack copes better with both the smooth and abrupt motion. At every kth iteration, the quality of the estimated position upon convergence (global best) is evaluated using its fitness value. $f(gBest_k^g) \rightarrow 1.0$ indicates high likelihood whereas $f(gBest_k^g) \rightarrow 0$ indicates low likelihood or no similarity between an estimation and target.

When $f(gBest_k^g) \leqslant T_{MinF}$, where T_{MinF} is a threshold, we know that there is low resemblance between the estimation and target and most likely the proposal distribution may not match the actual posterior. Thus in this scenario, $\mathcal{E}$ is increased alongside the maximum number of iterations, K by an empirically determined step sizes m *and* n respectively. This drives the swarm of particles to explore the region beyond the current local maxima (increase exploration). However, when an object has left the scene, K tend to increase continuously and cause swarm explosion. Thus, we limit $K \subset \mathcal{S}$.

$$\mathcal{E} \propto f(gBest_k^g) \tag{5}$$

In another scenario, where $f(gBest_k^g) \geqslant T_{MinF}, \mathcal{E}$ is decreased alongside K; constraining the search around the current local maximum (exploitation). In a straightforward manner, it is always best to drive particles at its maximum velocity to provide a reasonable bound in order to cope with the maximum motion change. However, this is not reasonable for real-time applications as it incurs unnecessary computational cost especially when the motion is not abrupt. Thus, by introducing the adaptive scheme to automatically adjust the exploration and exploitation behaviour of the swarm, SwATrack is able to cope with both the smooth and abrupt motion with reduced computational cost. Also, we observed that since the particles in SwATrack exchange information with one another, a minimal number of particles is sufficient for sampling.

4.2. Novel velocity model

With the introduction of DAP and $\mathcal{E}$, the novel velocity model, $\grave{v}$ in our PSO framework is written as:

$$\grave{v}_{k+1}^i = \mathcal{E}_k\left[(\omega * \grave{v}_k^i) + \left(c_1 * r_1 * \left(pBest_k^i - x_k^i\right)\right) + (c_2 * r_2 * (gBest_k - x_k^i))\right] \tag{6}$$

where, $\mathcal{E}_k$ is the exploration factor at iteration k, and c, r, ω are the acceleration parameters with the condition $p(w \cap c_1 \cap c_2) = 1.0$. The normalised condition applied to the acceleration allows on the fly tuning of these parameters according to the quality of the fitness function. The fitness function used here is represented by the normalised distant measure between the appearance model of an estimation and the object-of-interest. The fitness value of a particle, $f(x_k^i)$ measures how well an estimation of the object's position matches the actual object-of-interest; where 1.0 represents the highest similarity between an estimation and target and 0 represents no similarity.

At every kth iteration, each particle varies its velocity according to Eq. (6) and move its position in the search space according to:

$$p(x_{k+1}^i) = p(x_k)^i + \grave{v}_{k+1}^i \tag{7}$$

Note that the motion of each particle is directed towards the promising region found by the global best, $gBest_k$ from previous iteration, $k = k - 1$. The proposed SwATrack algorithm is shown in Algorithm 2.

Algorithm 2. Proposed SwATrack

Initialisation, at iteration $k = 0$

- Initialise a population of I particles, $\{x_k^i\}_{i=1,\ldots,I}$ with positions, $p(x_k^i)$, at random within the search space, $\mathcal{S}$.
- Initialise the velocities, $v(x_k^i)$ at random within $[1, -1]$.
- Evaluate the fitness value of each particle and identify their personal best $pBest_k^i = f(p(x_k^i))$.
- Identify the global best gth particle and update the global best information, $gBest_k = f(p(x_k^g))$.

for $k = 1$ to K **do**

  for $i = 1$ to I **do**

    Compute the new velocity according to:

$$\grave{v}_{k+1}^i = \mathcal{E}_k\left[(\omega * \grave{v}_k^i) + \left(c_1 * r_1 * \left(pBest_k^i - x_k^i\right)\right) + (c_2 * r_2 * (gBest_k - x_k^i)\right]$$

where,

$$p(w \cap c_1 \cap c_2) = 1$$

if $f(gBest_k^g) \leqslant T_{MinF}$ **then**
$\mathcal{E} = \mathcal{E} + n, K = K + n$
else
$\mathcal{E} = \mathcal{E} - n, K = K - n$
end if

if $\mathcal{C} \to 1$ **then**
$c_1 = c_1 + m, c_2 = c_2 - m, \omega = \omega + m$
else
$c_1 = c_1 - m, c_2 = c_2 + m, \omega = \omega - m$
end if
Update the position according to:

$$p(x_{k+1}^i) = p(x_k)^i + \grave{v}_{k+1}^i \tag{8}$$

Check for out of bound:

$$p(x_{k+1}^i) \subset \mathcal{S} \tag{9}$$

Update variables, $pBest_k^i, p\left(pBest_k^i\right), g, gBest_k, p(gBest_k^g)$
Check for **Convergence**

if Convergence == TRUE **then**
Terminate iteration
else
Continue
end if

end for
end for

5. Experimental results and discussion

In this section, we verify the feasibility and robustness of our proposed method in handling abrupt motion using a machine with a configuration of Intel core-i7, 2. 1 GHz with 8 GB RAM. The proposed SwATrack was implemented with C++ and OpenCV library.

5.1. Experimental settings

We assumed that the object-of-interest is known and hence manually initialise the 2D position of the target in the first frame. Automatic initialisation of target is a challenging research topic itself, and thus is not the in the scope of this study. The object is represented by its appearance model, which comprises HSV histogram with uniform binning; 32 bins. The normalised Bhattacharyya distant measure is used as the fitness value (cost function) to measure the quality of the estimation; where 1 represents the highest similarity between an estimation and target and 0 represents no similarity. Here, the initial values for SwATrack are $\mathcal{E} = 25, \omega = 0.4, c_1 = 0.3, c_2 = 0.3, K = 30, I = 15$ respectively. These values are set empirically and are not as critical; the adaptive mechanism in the proposed method allows adjustment of these parameters according to the quality of the observation model.

We compare the state-of-the-art results of PSO, PF [35,20], BDM [33], FragTrack [1], A-WLMC [12] and CT [39], respectively in terms of both the detection accuracy (%) and processing time (milliseconds per frame). In all experiments, the parameters of the state-of-the-art algorithms are fine-tuned accordingly (to the best understanding of the authors) for fair comparison. We reimplemented the standard PSO and PF [35,20] methods with C++ and OpenCV library, while the rest have been shared by the respective authors. Only the BDM tracker in[33] was implemented using the Matlab Image Processing Toolbox while the others are in C++ and OpenCV library.

5.2. Dataset

The proposed SwATrack was tested with our newly introduced abrupt motion dataset – namely the **M**alaya **A**brupt **Mo**tion *(MAMo)* dataset. This dataset comprises 12 videos as illustrated in Fig. 3 and Table 1. These sequences are arranged according to the different challenging scenarios as described in the following:

(a) Rapid motion of small object: There are 5 video sequences in this scenario to test the effectiveness of the proposed method in terms of tracking small object (e.g. table tennis ball) that exhibits fast motion. $TableT_1$ is the SIF Table Tennis sequence – a widely used dataset in the area of computer vision, especially for evaluation of detection and tracking methods [33]. This sequence has complex, highly textured background and exhibit camera movement with some occlusion between the ball and the player's arm. $TableT_2$ is a sample training video from the ITTF video library which is created to expose players, coaches and umpires to issues related to service action. Although this sequence is positioned to provide the umpire's point of view of a service, it is very challenging as the size of the tennis ball is very small; about 8x8 pixels to 15x15 pixels for an image resolution of 352 x 240. The video comprises of 90 frames including 10 frames in which severe occlusion happens, where the ball is hidden by the player's arm. $TableT_3$ is a match obtained from a publicly available source. In this sequence, the tennis ball is relatively large as it features a close-up view of the player. However, there are several frames where the ball appears to be blurred due to the low frame rate and abrupt motion of the tennis ball. $TableT_4$ and $TableT_5$ were captured at a higher frame rate, thus the spatial displacement of the ball from one frame to another appears to be smaller (less abrupt) and the ball is clearer. This is to test the ability of proposed method to handle normal visual tracking scenario. Since the ground truth for these data were not provided, we manually labelled the ground truth of nth object in each sequence. The ground truth is described as bounding box information, $X^n(x^n,y^n,w^n,h^n$ = *positions in the x-dimension, y-dimension, width, height*).

(b) Switching camera: In general, the sampling-based methods often assume a large variance in the proposal density to cope with abrupt motion. However, a large variance tends to decrease the tracking accuracy when tracking smooth motion. Thus, we include the scenario of tracking using sequences obtained from switching between multiple cameras to evaluate the tracking methods in coping with both, the abrupt and smooth motion. This category comprises 3 videos which includes the (Youngki, Boxing) and $Malaya_1$ sequences. The *Youngki* and *Boxing* can be found at [14], where they consist of frames edited from changes of camera shots between multiple cameras, where the hand-over between cameras are aimed at tracking a particular object throughout the scene. Due to the object's handover between multiple cameras, the object appears to have drastic change in position between adjacent frames during the switching period as well as the scale. Otherwise, the object exhibits smooth motion as it is captured by a single camera. The $Malaya_1$ sequence is created by combining the frames in the *Boxing* and *Youngki* sequences in an alternative manner. This combination is done to introduce definite tracking error when tracking the boxer in the *Boxing* sequence; since the boxer is missing in the *Youngki*

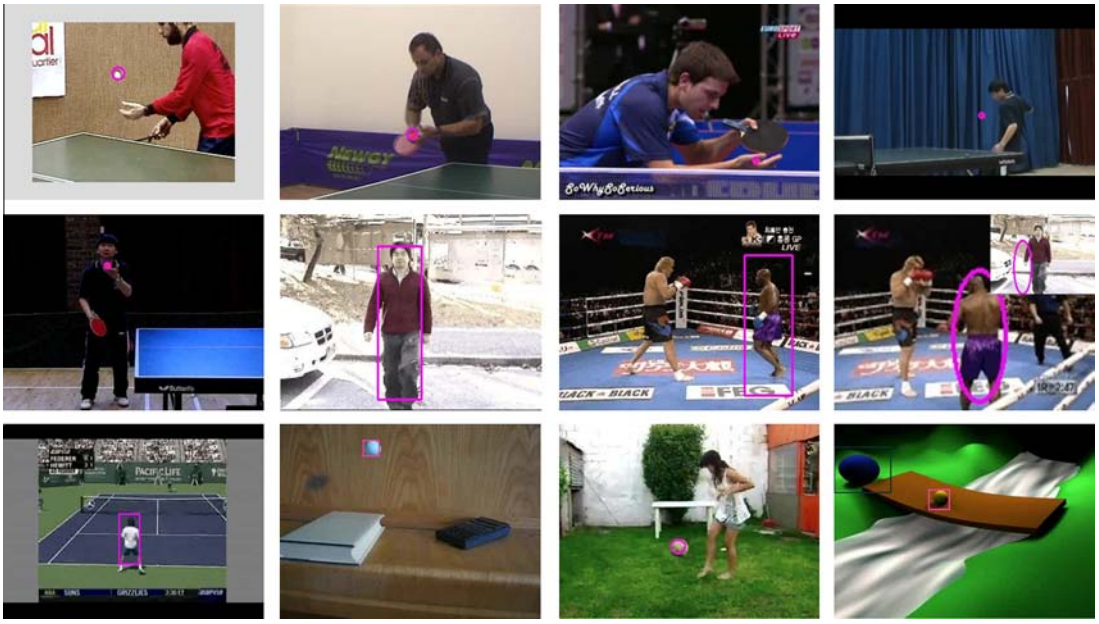


Fig. 3. Sample shots of the newly introduced Malaya Abrupt Motion (MAMo) dataset. This collection of data comprises 12 videos which exhibit the various challenging scenarios of abrupt motion. Top row, from left to right: $TableT_1$,$TableT_2$,$TableT_3$,$TableT_4$; Second row, from left to right: $TableT_5$, *Youngki*, *Boxing*, $Malaya_1$; Bottom row, from left to right: *Tennis*, $Malaya_2$, $Malaya_3$, $Malaya_4$.

Table 1
Summary of the Malaya Abrupt Motion *(MAMo)* dataset.

	Category	Name of the video sequences	Number of videos
(a)	Rapid motion of small object	$TableT_{1-5}$	5
(b)	Switching camera	*Youngki*, *Boxing* and $Malaya_1$	3
(c)	Partially low-frame rate	*Tennis*	1
(d)	Inconsistent speed	$Malaya_{2-3}$	2
(e)	Multiple targets	$Malaya_4$	1
	Total		**12**

sequence. The simulation of inaccurate tracking scenario is to test the robustness of tracking methods in not only tracking abrupt motion, but recovery from inaccurate tracking.

(c) Partially low frame rate: In another example of abrupt motion, we simulate the scenario of tracking in partially low frame rate. The *Tennis* video [14] comprises of downsampled data to simulate abrupt change caused by low frame rate. The frames are downsampled from a video with more than 700 original frames, by keeping one frame in every 25 frames. The rapid motion of the tennis player from one frame to another due to the downsampling made tracking extremely difficult. Downsampling is done to simulate abrupt motion during low-frame rates.

(d) Inconsistent speed: We obtained 2 videos from the YouTube, where each sequence comprises an object which moves with inconsistent speed throughout the sequence. The first video, $Malaya_2$, aims to track a synthetic ball which moves randomly across the sequence with inconsistent speed, whilst the second video, $Malaya_3$ tracks a soccer ball which is being juggled in a free-style manner in a moving scene with a highly textured background (grass).

(e) Multiple targets: This is to demonstrate the capability of the proposed system to track multiple targets; whilst most of the existing solutions are focused on single target. We created a synthetic video, $Malaya_4$ that consists of two simulated balls moving at random speed.

In general, most of the video sequences in *MAMo* dataset are well diversified as most of them contain a mixture of both the smooth and abrupt motion. It is less likely for an object-of-interest to move with abrupt motion at all time, unless the video is captured at low frame rate as exhibit by the *Tennis* sequence, in particular. The *MAMo* dataset is publicly available along with their corresponding ground truth information.[1]

5.3. Quantitative results

5.3.1. Experiment 1: Detection rate

Detection rate refers to the correct number and placement of the objects in the scene. For this purpose, we denote the ground truth of nth object as GT_n, and the output from the tracking algorithms of jth object is denoted as, ξ_n. We describe the ground truth and tracker output of each nth object as bounding box information, $X^n(x^n,y^n,w^n,h^n$ = *x-position, y-position, width, height)*. The coverage metric determines if a *GT* is being tracked, or if an ξ is tracking accurately. In [24], it is shown that the *F-measure*, *F*, suited this task as the measure is 1.0 when the estimate, ξ_n overlaps perfectly with the ground truth, GT_n. Two fundamental measures known as *precision* and *recall* are used to determine the *F-measure*.

5.3.1.1. Recall. Recall measures how much of the *GT* is covered by the ξ, and takes value of 0 if there is no overlap and 1.0 if the estimated position fully overlap with the actual locality of the target. Given a ground truth, GT_n, and a tracking estimate, ξ_n, the *recall*, $\Re_n$ is expressed as:

$$\Re_n = \frac{|\xi_n \cap GT_n|}{|GT_n|} \tag{10}$$

5.3.1.2. Precision. Precision measures how much of the ξ covers the *GT* takes value of 0 if there is no overlap and 1 if they are fully overlapped. The *precision*, $\wp_n$ is expressed as:

$$\wp_n = \frac{|\xi_n \cap GT_j|}{|\xi_n|} \tag{11}$$

5.3.1.3. F-measure. The *F-measure*, F_n is expressed as:

$$F_n = \frac{2\Re_n\wp_n}{\Re_n + \wp_n} \tag{12}$$

[1] http://web.fsktm.um.edu.my/~cschan/project3.htm.

Table 2
Experiment results – comparison of the detection rate (in %).

	PSO	PF [35,20]	BDM [33]	FragTrack [1]	A-WLMC [12]	CT [39]	SwATrack
TableT1	70.1	58.4	68.3	64.9	47.2	72.3	**87.8**
TableT2	83.1	69.8	53.4	24.1	3.2	4.3	**93.1**
TableT3	58.2	52.1	67.3	55.3	8.7	24.5	**74.1**
TableT4	59.6	47.3	73.2	57.2	6.9	**98.2**	97.3
TableT5	60.3	34.5	64.2	9.7	5.4	36.3	**72.8**
Average	66.26	52.42	65.28	42.24	14.28	47.12	**85.02**

Coverage test: In this experiment, we employ the *F-measure* according to the score measurement of the known PASCAL challenge [9]. That is, if the F_n of nth object is larger than 0.5, the estimation is considered as correctly tracked in the frame. Tables 2 and 3 demonstrates the detection accuracy of the benchmarked tracking algorithms for all 8 test sequences. Overall, the experimental results show that the average tracking accuracy of the proposed method surpasses most of the state-of-the art tracking methods with an average detection accuracy of 91.39%. For all 6 test sequences ($TableT_1, TableT_2, TableT_5$, *Youngki* and *Tennis*), the SwATrack generates the best tracking results amongst the rest and ranked second best for sequence $TableT_4$ and *Boxing*, respectively.

Methods that are not built based on sophisticated motion model such as the FragTrack [1] performs poorly, overall with an average accuracy of 37.19%. Their method, which employs a refined appearance model that adapts to the changes of the object, copes well with partial occlusion. However, it is still dependent on the search radius and thus fails when tracking abrupt motion, where the object tends to be outside the search window. PF on the other hand, achieves a detection accuracy of 85.6%. This is expected, since the PF algorithm is constrained to a fixed Gaussian motion model. Once PF has lost track of the object, it has the tendency to continue searching for the object in the wrong region; leading to error propagation and inability to recover from incorrect tracking such as shown in Fig. 4(a). Fig. 4(a) demonstrates sample shots of scenarios of abrupt motion, where the PF tracker exhibits the state of being trapped in local optima. At frame 449–451, the PF tracker continues to locate the object within the assumed Gaussian distribution when the object has in actual fact, moved abruptly to the other corner of the image. On the other hand, the proposed SwATrack copes better with abrupt motion and is not likely to get trapped in local optima; since the exploitation and exploration is self-adjusted based on the fitness function, and is as shown in Fig. 4(b). Thus, as shown in Fig. 4(b), the SwATrack is able to track the object accurately although the motion is highly abrupt. Similarly, the inability of MCMC and its variants, the A-WLMC [12] and IA-MCMC [41] tracking methods in handling abrupt motion is as shown in Fig. 5.

Dataset unbias: The problem of dataset bias was highlighted in [27] where the paper argue that *"Is it to be expected that when training on one dataset and testing on another there is a big drop in performance?"*. Motivated by this, we replicate similar scenario in the tracking domain, and observe that although the A-WLMC method [12] performs well in $TableT_4$ and *Youngki* sequence, they do not produce consistent results when tested across the other datasets as shown in Tables 2 and 3. For example, we notice that the accuracy of A-WLMC changes drastically from one tracking scenario to another, with an average detection accuracy for *TableT* video sequences is fairly low at, 14.28% while for the *Tennis*, *Boxing* and *Youngki* sequences, it performs remarkably well with an average accuracy of 93.33%. This provides an indication that the A-WLMC solution [12] maybe suffers from dataset bias problem, as it seems to only work well in their proposed dataset, but performed poorly when it is employed on different sequences. Perhaps this is due to the motion model employed by these tracking methods that works well only on certain scenarios, alluding to the notion in [10] that *different motion requires different motion models*. This is indeed not the case for our proposed SwATrack. Our overall detection rates are 85.02% and 97.76%, respectively. For all sequences that exhibit different challenging conditions, e.g. rapid motion ($TableT_{1-5}$), camera switching (*Youngki* and *Boxing*), low-frame rate (*Tennis*), the SwATrack has shown its ability to cope with the various scenarios of abrupt motion.

Size invariance: We further investigated the dataset bias problem and found out that there is an influence of the object size to the detection rate. For instance, the A-WLMC algorithm [12] performs poorly for sequences in which the resolution of the object-of-interest is relatively small, such as in the *TableT* video sequences and performs surprisingly well when the object is large such as in the *Youngki*, *Boxing* and *Tennis* sequences. This indicates the need to have better representation of the object for a more accurate acceptance and rejection of estimations in the MCMC algorithm.

Table 3
Experiment results – comparison of the detection rate (in %).

	PSO	PF [35,20]	FragTrack [1]	A-WLMC [12]	SwATrack
Tennis	87.3	67.3	20.6	95.1	98.3
Youngki	87.1	47.2	27.5	86.8	98.7
Boxing	82.4	16.3	48.3	98.1	96.3
Average	85.6	43.6	32.13	93.33	**97.76**

(a) Sample detections from PF tracking.

(b) Sample detections from SwATrack tracking.

Fig. 4. Sample output to demonstrate the incorrect tracking state, which is caused by trapped in local optima. The aim of this sequence is to track the person in dark skin and purple short. From Frame 449–451 (a), PF lost track of the object due to sampling from incorrect distribution during abrupt motion. Thus, it can be observed that PF continues to track the object inaccurately once it has lost track of the object. On the other hand, the results in (b) demonstrate the capability of the SwATrack tracker in dealing with the non-linear and non-Gaussian motion of the object (Best view in colour). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

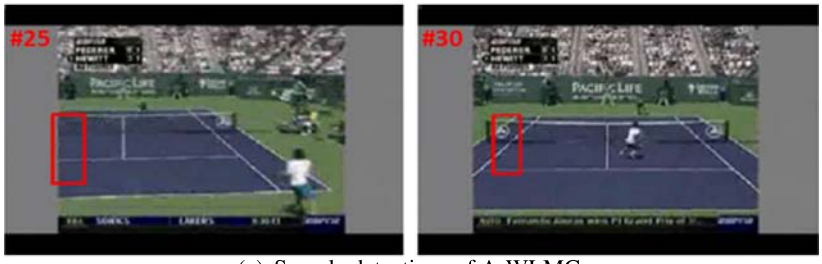
5.3.2. Experiment 2: Computational cost

Fig. 6 demonstrates the comparison results between the proposed method and the state-of-the-art tracking solutions in terms of time complexity. It is observed that the SwATrack algorithm requires the least processing time with an average of 63 ms per frame. On the contrary, the MCMC-based solutions which include the A-WLMC [12] and PF [35,20] require higher processing time. This is likely due to the inherent correlation between the MCMC samplers which is known to suffer from slow convergence when an object has not been tracked accurately. In our experiments, we notice that in scenarios where the MCMC requires high processing time, the accuracy of the MCMC is minimal. The increase in computational cost is due to the increase of search space when the observation model is unlikely representing the object. Note that the optimal number of samples deployed in the PF and MCMC throughout the sequences has been selected empirically; where it ranges from 150 to 1000 particles in PF, 600 to 1000 particles in MCMC with 600 iterations while the SwATrack uses 10–50 particles (15$\times$ in reduction) with 5–70 iterations. Intuitively, an increase in the number of samples would lead to an increase in computational cost as each particle would need to be evaluated against the appearance observation; explaining the minimal processing time required by the proposed SwATrack.

As shown in Tables 2 and 3, in which the SwATrack detection rate is ranked second, we observe that although the CT [39] and A-WLMC [12] achieved better accuracy, their average processing time are threefold as compared to the SwATrack. This is due to the need to increase the subregions for sampling when the state space increases in the A-WLMC algorithm [12]. On the contrary, our method adaptively increases and decreases its proposal variance for a more effective use of samples. Thus the processing time required is much lower as compared to the other methods. The advantage of the dynamic mechanism is reflected when comparing the processing time of SwATrack to standard PSO (average of 195.20 ms per frame); where the processing time of PSO is three times greater than that of the SwATrack. In summary, the experimental results demonstrate the capability of the proposed system to cope with the variety of scenarios which exhibit highly abrupt motion. The adaptation of a stochastic optimisation method into tracking abrupt motion has been observed to incur a slight increase in the

(a) Sample detections of PF.

(b) Sample detections of SwATrack.



(c) Sample detections of A-WLMC.

(d) Sample detections of IA-MCMC.

Fig. 5. A comparison between PF, SwATrack, A-WLMC [14] and IA-MCMC [41]. It is observed that the SwATrack tracking gives a more accurate fit of the object's locality.

processing cost, yet at the same time is able to have fair tracking accuracy as compared to the more sophisticated methods. Thus, the preliminary results at this stage, gives a promising indication that sophisticated tracking methods may not be necessary after all.

5.4. Qualitative results

5.4.1. Partially low frame rate

The sequence aims to track a tennis player in a low-frame rate video, which has been downsampled from a 700 frames sequence by keeping one frame in every 20 frames. Here, the object (player) exhibits frequent abrupt changes which violate the smooth motion and constant velocity assumptions. Thus, motion that is governed by Gaussian distribution based on the Brownian or constant-velocity motion models will not work in this case. Fig. 5 shows sample shots to compare the performance between the PF tracking (500 samples), A-WLMC (600 samples) [14], IA-MCMC (300 samples) [41] and SwATrack (50

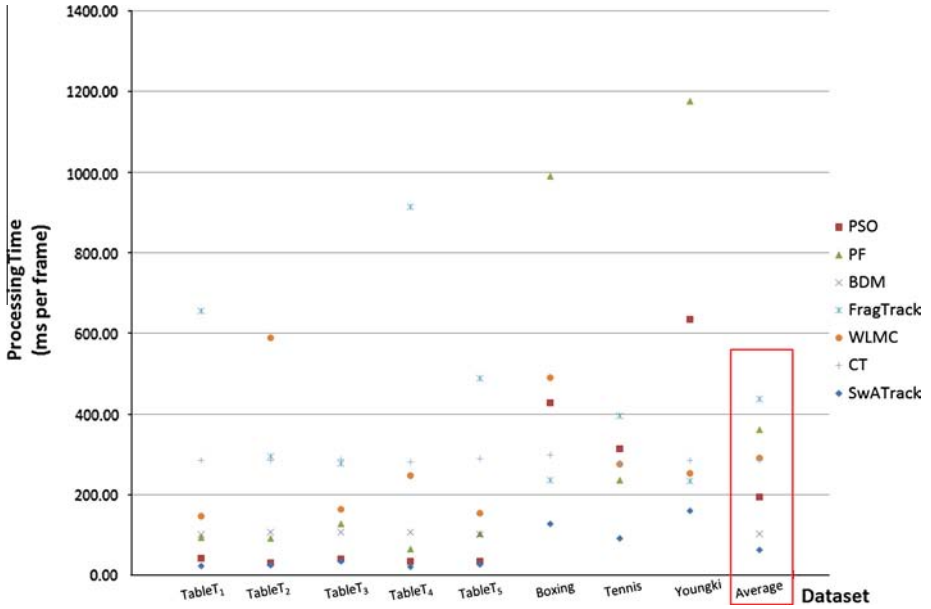


Fig. 6. Time Complexity. This figure illustrates the comparison in terms of processing time (milliseconds per frame) between the proposed SwATrack, standard PSO, PF, BDM, FragTrack, A-WLMC [12] and CT.

samples). It is observed that the tracking accuracy of SwATrack is better than PF and A-WLMC [14] even by using fewer samples. While the performance of SwATrack is comparable to IA-MCMC [41], SwATrack requires fewer samples and thus requires less processing requirement. These results further validated that the proposed SwATrack is able to track moving object accurately and effectively, regardless of the variety of change in the object's motion.

5.4.2. Local minimum problem

In this experiment, we aim to test the capability of SwATrack to recover from incorrect tracking. This experiment would in particular, evaluate the efficacy of the proposed DAP and $\mathcal{E}$ in handling abrupt motion. Fig. 7 shows the result for *Youngki* and *Boxing* sequences, which exhibits abrupt motion in a camera switching scenario. Due to the object's handover between multiple cameras, the object appears to have drastic change in position between adjacent frames during the switching period as well as the scale. Otherwise, the object exhibits smooth motion as it is captured by a single camera. The switching happens repeatedly when an object moves out of a particular camera view, into the field of view of another camera. The assortment of both smooth and abrupt motion would test the capability of the proposed DAP in increasing its exploitation during smooth motion and increasing its exploration during abrupt motion. In Fig. 7(a), it is observed that during the switch from frame 247

(a) Sample of SwATrack on *Boxing* sequence.

(b) Sample of SwATrack on *Youngki* sequence.

Fig. 7. Sample outputs to demonstrate the flexibility of the proposed SwATrack to recover from incorrect tracking. It can be noticed that the SwATrack only requires minimal frames (1–2 frames) to escape from local optima and achieve global maximum.

and 248, the SwATrack appears to have inaccurate tracking of the object; as the estimated position which is highlighted by the ellipse does not overlap accurately with the object. However, due to the flexibility of the proposed DAP and $\mathcal{E}$ which allows self-adjustment of the exploitation and exploration of the swarm based on the fitness function, the SwATrack algorithm is able to recover from the inaccurate tracking within minimal number of frames. Similar behaviour is observed in the *Youngki* sequence which further validated our notion.

Nonetheless, we simulated another challenging scenario – the $Malaya_1$ in which incorrect tracking is most likely to happen by sampling frames from 2 different datasets as shown in Fig. 8(a). The frames in the *Boxing* sequence are combined in an alternative manner with the frames from the *Youngki* sequence. In this combined sequence, the object-of-interest which is highlighted in the ellipse in Frame 1 of in Fig. 8(a) tend to disappear from one frame and re-appear in the subsequent frame interchangeably. From the qualitative results shown in Fig. 8(b), we observe that the A-WLMC tracking [14] is not robust and does not cope well with inaccurate tracking. When the object-of-interest disappear from the scene (i.e. Frame 77), the A-WLMC [12] gives an erroneous estimation of the object. In the subsequent frame, where the object re-appears, the A-WLMC [12] has difficulty recovering from its tracking such as shown in Frame 78 where the estimation does not fit the actual position of the object accurately. In the subsequent frames, the A-WLMC [12] tend to continuously missed tracked of the object. Although the sampling efficiency in the A-WLMC [12] adopts a more efficient proposal distribution as compared to the standard PF, it is still subjected to a certain degree of trapped in local optima. Furthermore, the A-WLMC [12] utilizes the information of historical samples for intensive adaptation, thus requiring more frames information to recover from inaccurate tracking. The proposed SwATrack on the other hand, is observed to work well in this $Malaya_1$ video sequence, where minimal frame is required to recover from erroneous tracking. As shown in Fig. 8(c), the SwATrack is able to track the object accurately when the object appears or re-appears in the scene (as shown in the even frame number). This is made possible due to the information exchange and cooperation between particles in a swarm that provide a way to escape the local optima and reach the global maximum; leading to and optimised proposal distribution.

5.4.3. Swarm explosion problem

In the conventional PSO algorithm, the lack of a mechanism to control the acceleration parameters and the dependency on randomness in the system fosters the danger of swarm explosion and divergence. When swarm explosion or divergence happens, the velocities and positions of each particle are steered towards infinity and thus, preventing convergence. In the context of our study, swarm explosion and divergence is very likely. This is due to the tendency of the swarm to increase its exploration in order to deal with the abrupt change in an object locality. Thus, in this combination sequences (similar sequence as shown in Fig. 8) where the boxer disappears and reappears in the scene from one frame to another, we observe that the conventional PSO fail to track the abrupt motion of the boxer accurately as shown in Fig. 9(a). When the object disappears from the scene (since the boxer is missing in the *Youngki* sequence), the swarm tends to increase its exploration and is most likely to steer towards infinity; explosion happens. If this happens, the swarm lose track of the object and is most likely to continue searching from an inaccurate distribution leading to continuous inaccurate tracking of the object. However, in the proposed SwATrack, recovery from incorrect tracking is made possible by the Dynamic Acceleration Parameters

(a) Sample shots of the dataset that is obtained by combining frames from two different sequences. The object enclosed in the ellipse is the object to be tracked.

(b) Sample detections by the A-WLMC tracking. A-WLMC tend to tracked the object inaccurately once it has lost or missed tracked of the object as shown from Frame 79 onwards.

(c) Sample detections by the SwATrack tracking. In Frame 77, since the object-of-interest does not appear in the frame, inaccurate tracking happens. However, the SwATrack is able to recover its tracking at the following frame, Frame 78.

Fig. 8. Sample outputs to demonstrate the capability to recover from incorrect tracking.

(a) Sample detections by the conventional PSO tracking. PSO tracker tend to tracked the object inaccurately once it has lost or missed tracked of the object as shown from Frame 103 onwards.

(b) Sample detections by the SwATrack tracking. In Frame 103, since the object-of-interest does not appear in the frame, inaccurate tracking happens. However, the SwATrack is able to recover its tracking at the following frame, Frame 104.

Fig. 9. Sample outputs to demonstrate the inaccurate tracking in conventional PSO due to swarm explosion, and the capability of the proposed SwATrack to track object accurately.

(DAP) and Exploration Factor $\mathcal{E}$ mechanism, which prevents the particles from steering towards infinity by expanding and constricting the velocity of particles. See Fig. 9.

5.4.4. Invariant to object size

We further tested the proposed SwATrack, PF [35,20] and A-WLMC [14] on resized sequences of similar set of datasets to simulate the scenario in which the object size is smaller. Thus, the initial frame size of 360x240 is reduced into half, to 180x120 pixels. From our observations, the SwATrack is the least sensitive towards the size of object-of-interest, while the detection accuracy of the A-WLMC is reduced as the size of object gets smaller. This is due to the robustness of the optimised sampling in SwATrack as compared to the least robust method of rejection and acceptance as proposed in the A-WLMC. The overall detection accuracy of the proposed SwATrack remain at an average of 90% regardless of the object's size whereas the detection accuracy of PF and A-WLMC decrease significantly by more than 25% when the object's size decreases. Sample output is as shown in Fig. 10. Finally, we evaluated the proposed SwATrack on videos obtained from Youtube ($Malaya_{2-4}$); and the qualitative results are as depicted in Fig. 11. It is observed that the SwATrack is able to track the abrupt motion of the balls efficiently, as well as the capability of the proposed system to track multiple objects; two simulated balls moving at random. From the best of our knowledge, most of the existing solutions [14,41] are focused on single object.

6. Discussion

6.1. Will increasing the complexity of tracking algorithms enhance better results in abrupt motion tracking?

Motivated by the meta-level question prompted in [42] on *whether there is a need to have more training data or better models for object detection*, we raise similar question in the domain of this area; will continued progress in visual tracking be driven by the increased complexity of tracking algorithms? Intuitively, an increase in the number of samples in sampling-based tracking methods such as PF and MCMC would increase the tracking accuracy. One may also argue that the additional computational cost incurred in the iterative nature of the proposed SwATrack and MCMC would complement the higher number of particles required by the PF. Thus, in order to investigate if these intuitions hold true, we perform experiments using an increasing number of samples and iterations *(Sampling-based vs. Iterative-based solutions)*. We then observe the behaviours of PF and SwATrack in terms of accuracy and processing time with the increase in complexity. PF is chosen in this testing as it bears close resemblance to the proposed SwATrack algorithm in which a swarm of particles are deployed for tracking. The experiments in this section are performed on the $TableT_{1-5}$ video sequences.

6.1.1. Number of particles vs. accuracy

6.1.1.1. Particle filter. In the PF algorithm, we vary the number of samples or particles (i.e. 50, 100, . . ., 2000) used throughout the sequence to determine the statistical relationship between number of samples and performance. We gauge the performance by the detection accuracy (%) and processing time (in milliseconds per frame). The average performance across all 5 *TableT* video sequences is as shown in Fig. 12(a). Sample of the performance for sequence $TableT_1$ and $TableT_2$ are as shown in Fig. 13(a).

The results demonstrate that the number of particles used in PF is correlated to the detection accuracy; where the growth in the number of particles tends to increase the accuracy. Similarly, the average time taken also increases exponentially as

(a) Sample detections from A-WLMC on reduced image size. A-WLMC has a high tendency to lose track of the object when it moves abruptly, and demonstrate continuous inaccurate tracking such as shown in Frame 279-284. Note that for similar frames, the A-WLMC tracker is able to track the object accurately when the image size is larger. Number of iterations = 600, particles = 600.

(b) Sample detections from SwATrack on reduced image size. SwATrack produces consistent tracking as compared to PF and A-WLMC, regardless of the size of object. Number of iterations = 30, particles =20.

Fig. 10. Qualitative Results: Comparison between A-WLMC and our proposed SwATrack in terms of reduced object size.

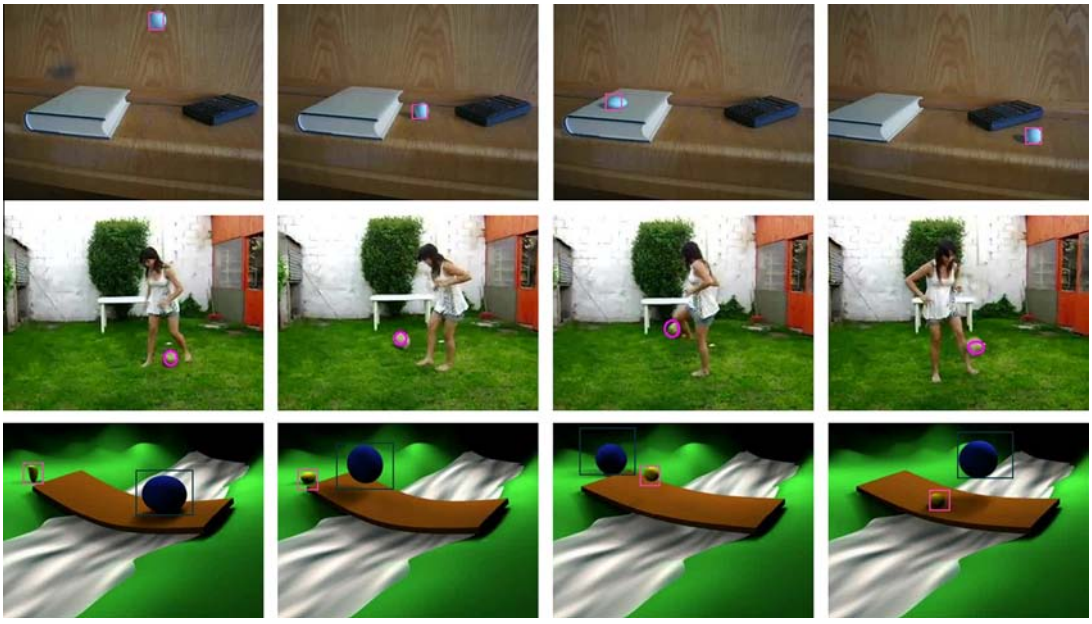
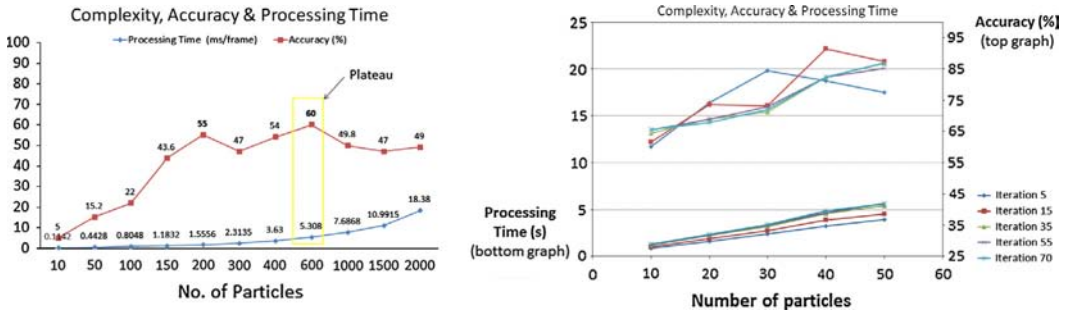


Fig. 11. Sample of SwATrack on tracking the object(s) in $Malaya_{2-4}$ video sequences.

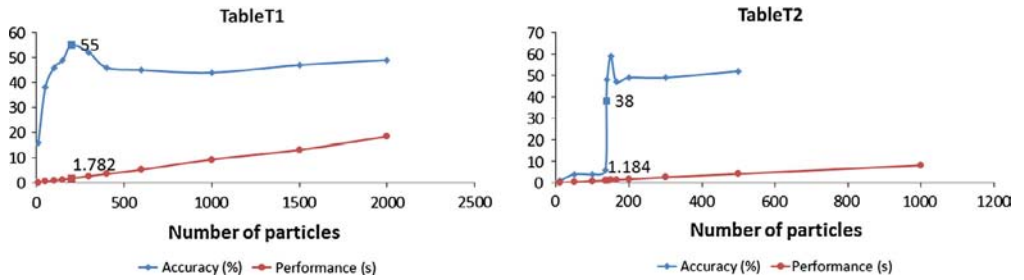
the number of particles used in PF grows. This alludes the fact that as the number of particles increases, the estimation processes which include object representation, prediction and update also multiply. However, it is observed that PF reaches a plateau after hitting the optimal accuracy, after which any increase in the number of particles will either have a decrease in accuracy or no significant improvement. From Fig. 12, we can see the detection accuracy decreases after the optimal solution, which is given when the number of particles is 600. Our findings instigate the underlying assumption that the increase of number of particles will lead to an increase in the accuracy. Thus, we raise the question of whether complex (in this context the complexity is proportional to the number of particles deployed) tracking methods are really necessary? Also, the best parameter configurations may differ from one sequence to another due to the different motion behaviour portrayed by the object in each sequence. For example, in Fig. 13(a), the optimal setting is 250 particles which produces detection accuracy of 55% and takes 1.78 s of processing time. Meanwhile, the second sequence has a different optimal setting of 150 particles as shown in Fig. 13(b). This advocates the notion as in [10] that *motion models indeed only work for sometimes*.

SwATrack: Similarly, we perform the different parameter settings test on the proposed SwATrack algorithm and the average results are demonstrated in Fig. 12(b), while Fig. 14 illustrate the results for $TableT_1$ and $TableT_2$. In addition to the number of particles used in PF tracking, the proposed SwATrack has an additional influencing parameter, the maximum



(a) PF (b) SwATrack

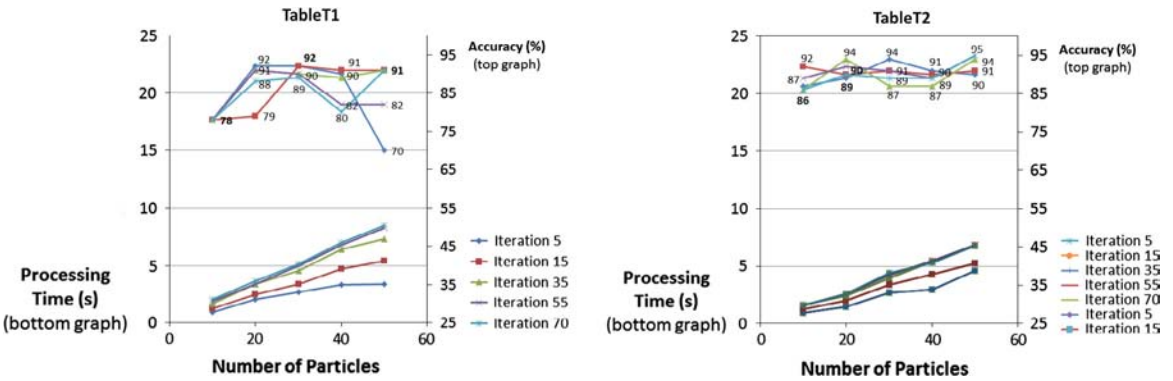
Fig. 12. A comparison in terms of accuracy vs. different number of samples and accuracy vs. different number of samples and iteration.



(a) Sequence $TableT_1$. (b) Sequence $TableT_2$.

Fig. 13. The accuracy and performance of PF with different number of samples for sequences, $TableT_{1-2}$.

number of iterations. We vary the number of particles against the number of iterations for fair evaluation. As illustrated in the left y-axis of the chart (bottom graph), we can see that the average processing time increases as the number of iterations increase. However, the processing time increases up to a maximum value; after which any increase in the iterations would not make much difference in its processing time. Notice that the processing time for larger numbers of iterations (55 & 70) tend to overlap with one another, demonstrating minimal increase in processing time as the number of iterations grows. This is due to the optimisation capability of the proposed SwATrack to terminate its search upon convergence, regardless of the defined number of iterations. This is particularly useful in ensuring efficient search for the optimal solution, with minimal number of particles. As for the detection accuracy, we can see that in general the average accuracy of the proposed SwATrack is higher than PF, with an average accuracy of 92.1% in the first sequence as shown in Fig. 14(a). The sudden decrease in accuracy for SwATrack tracking with 70 number of iterations as shown in Fig. 14(a) may be due to the erratic generation of random values in C++ implementation. This behaviour is not observed in other sequences, where their detection accuracy is



(a) Sequence $TableT_1$. (b) Sequence $TableT_1$.

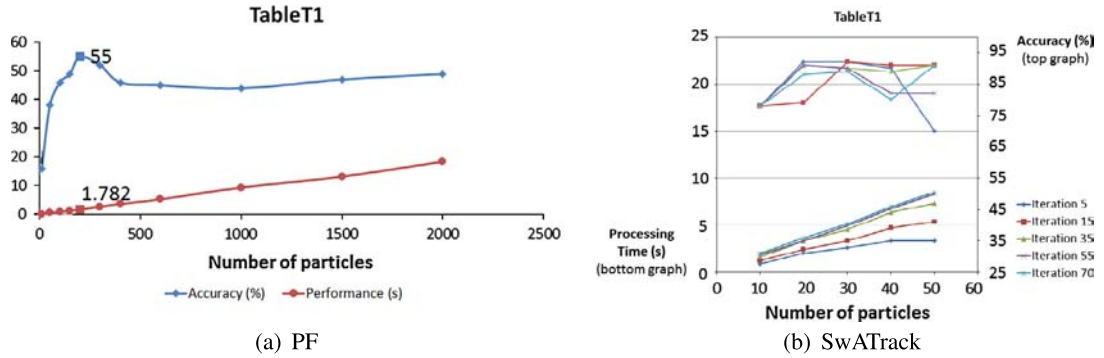
Fig. 14. The accuracy and performance of SwATrack with different number of samples and iterations for $TableT_{1-2}$.

consistent across frames. Thus far, we take an average result for each test case over 10 runs to ensure reliable results and we believe that with a higher number of runs, we would be able to obtain unbiased results without outliers. In summary, the results further validate our findings that the proposed SwATrack is able to achieve better accuracy as compared to PF, whilst requiring only about 10% of the number of samples used in PF with minimal number of iterations. This is made possible by an iterative search for the optimal proposal distribution, incorporating available observations rather than making strict assumptions on the motion of an object. Thus, we believe that the findings from our study create prospects for a new paradigm of object tracking. Again, we raise the question *if there is a need to make complex existing tracking methods by fusing different models and algorithms to improve tracking efficiency? Would simple optimisation methods be sufficient?*

6.1.2. Number of samples vs. processing time

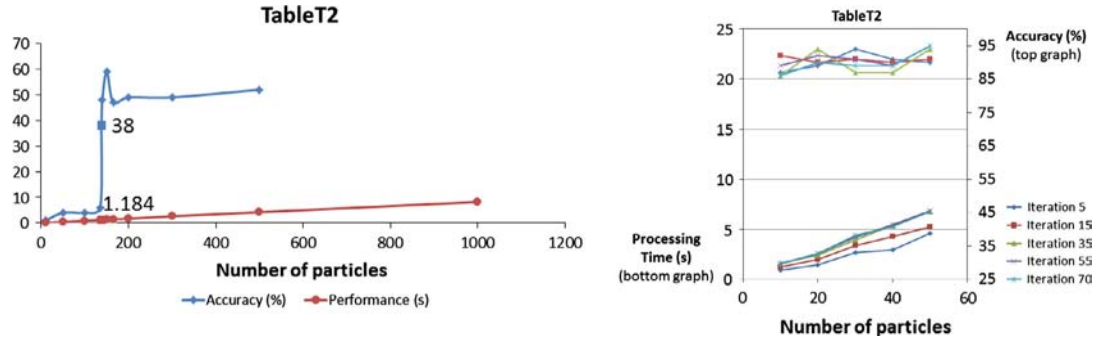
6.1.2.1. Particle filter. In the PF algorithm, we vary the number of samples or particles (i.e. 50, 100, . . ., 2000) used throughout the sequence to determine the statistical relationship between number of samples and detection accuracy. The lowest value of the parameter value is determined based on the minimal configuration to allow tracking while the highest value is set to the maximal configuration before it reaches a plateau detection accuracy. The detection accuracy and performance of the PF algorithm with different parameter settings are as shown in Figs. 15–19(a). The results demonstrate that the number of particles used in PF is related to the detection accuracy; where the increases in the number of particles tend to increase the accuracy. Similarly, the average time taken also increases as the number of particles used in PF grows. This alludes the fact that as the number of particles increase, the estimation processes which include object representation, prediction and update also multiply. However, it is observed that PF reaches a plateau detection accuracy after hitting the optimal accuracy, after which any increase in the number of particles will either have a decrease in accuracy or no significant improvement. Thus, the underlying assumption that the increase of number of particles will lead to an increase in the accuracy does not hold true. This may be due to the resampling step in most PF algorithms that is highly prone to error propagation. Also, the best parameter configurations may differ from one sequence to another due to the different motion behaviour portrayed by the object in each sequence. For example, in Fig. 15(a), the optimal setting is 250 number of particles which produces detection accuracy of 55% and takes 1.78 s of processing time. Note that in this set of experiments, other parameters such as the mean and variance for the Gaussian distribution in PF is not optimal values as compared to the earlier experiment. A standard configuration of Gaussian white noise is used across frames. Thus, the results obtained may slightly differ.

6.1.2.2. SwATrack. Similarly, we perform the different parameter settings test on the proposed SwATrack algorithm and the results are demonstrated in Figs. 15, 16, 17, 18 and 19(b). In addition to the number of particles used in PF tracking, the proposed SwATrack has an additional influencing parameter, the maximum number of iterations. Thus, here we vary the number of particles against the number of iterations and obtain their detection accuracy. As illustrated in the left y-axis of the chart (bottom graph), we can see that the average processing time increases as the number of iterations increase. However, the processing time taken reaches a maximal value, where the different number of iterations require almost comparable amount of time. This can be seen by the overlapping results as shown in Fig. 15–19(b), in particular. This demonstrate that the effectiveness of the termination criteria in the proposed SwATrack. When a global solution has been found by the entire swarm (swarm reaches convergence), the search activity terminates despite the initial setting value of the number of iterations. Also, our proposed method which automatically changes the number of iterations according to the swarm search quality allows a self-tuned setting of the maximum iteration number. As for the detection accuracy, we can see that in general the average accuracy of the proposed SwATrack is higher than PF, with an average accuracy of 92.1% in the first sequence as shown in Fig. 15(b). In summary, the results further validate our findings that the proposed SwATrack is able to achieve better accuracy as compared to PF, whilst requiring only about 10% of the number of samples used in PF with minimal number of iterations.



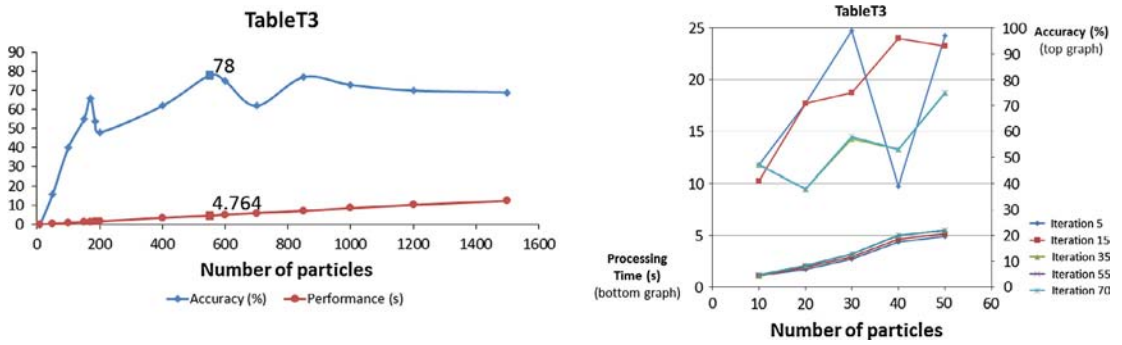
(a) PF (b) SwATrack

Fig. 15. *TableT*$_1$: The accuracy and performance of PF/SwATrack with different parameter settings.



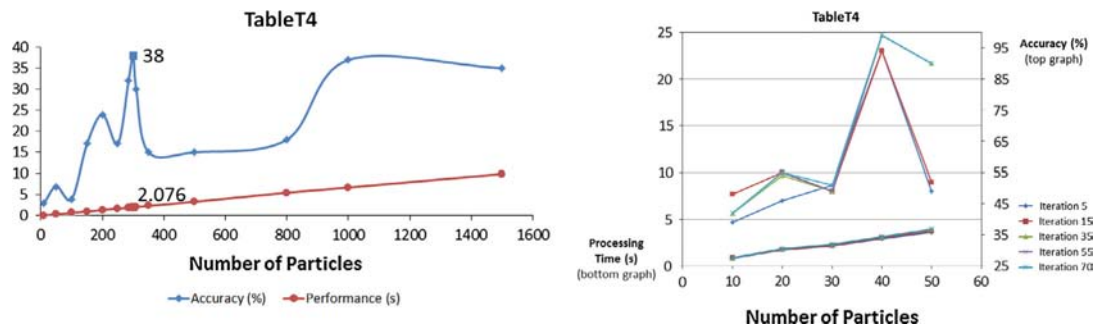
(a) PF (b) SwATrack

Fig. 16. *TableT*$_2$: The accuracy and performance of PF with different parameter settings.



(a) PF (b) SwATrack

Fig. 17. *TableT*$_3$: The accuracy and performance of PF with different parameter settings.



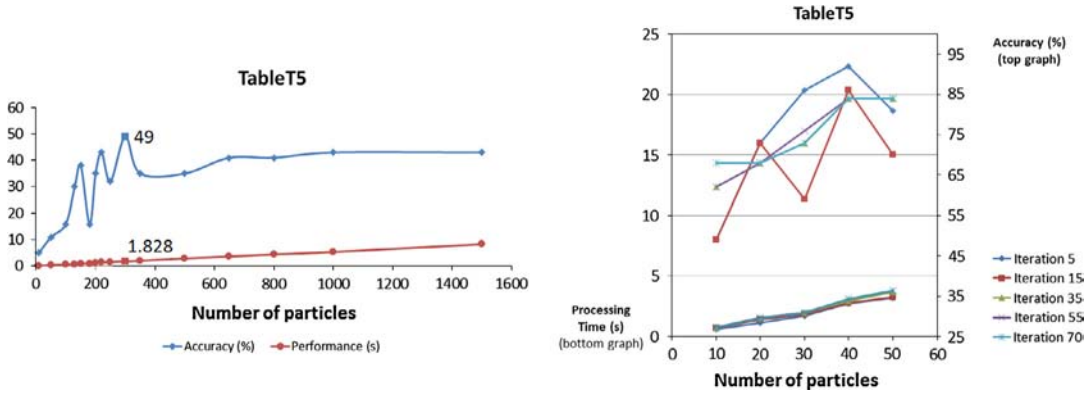
(a) PF (b) SwATrack

Fig. 18. *TableT*$_4$: The accuracy and performance of PF with different parameter settings.

6.1.3. Sampling strategy

To further evaluate the robustness of the proposed algorithm as well as to understand the behaviours of other algorithms when tracking abrupt motion, we perform the sampling strategy test. In this test, we simulate the scenario of receiving inputs from the sensors with a lower frame rate by downsampling the number of frames from the test sequence; assuming the actual data are obtained at normal rate of 25 frames per second to a lower rate of 5 frames per second. We named this new downsample *TableT* video sequences as the *DoS* − *TableT* and in total there are 4 video sequences. We have excluded the *TableT*$_5$ for this purpose, as the object appears to be out of the scene in the early frames of this video, and thus the downsampled sequence will comprise minimal number of frames in which the object appears in the scene.

Fig. 20 demonstrates the detection accuracy between the proposed SwATrack and PF for all four sequences by downsampling each sequence to simulate the 5 frames per second scenario. Note that the detection accuracy is determined by comparing the ground truth for the sampled frames only. However, it is observed that in general the proposed SwATrack



(a) PF (b) SwATrack

Fig. 19. *TableT*$_5$: The accuracy and performance of PF with different parameter settings.

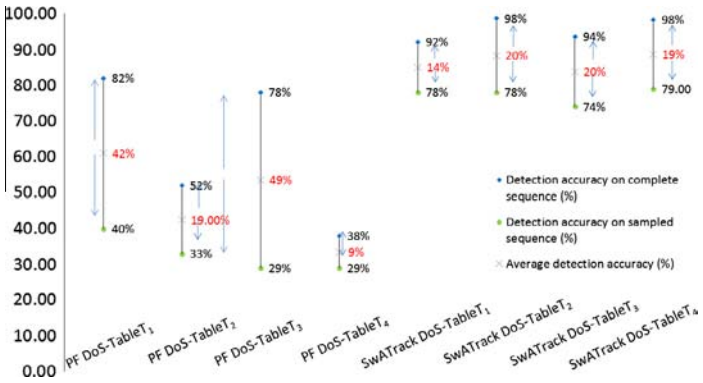


Fig. 20. The detection accuracy of SwATrack against PF during sampling for *DoS* – *TableT*$_{1-4}$.

has better detection accuracy as compared to PF in both situations; with and without sampling. The average detection accuracy of SwATrack for the complete sequences is approximately 95.5% whereas the average for PF is approximately 62.5%. During sampling, the average detection of accuracy of SwATrack is approximately 77.25% whereas PF is approximately 32.75%. We can see that the detection accuracy of PF drops drastically when the frame rate decreases. This is because, in low frame rate videos, the object tends to have abrupt motion and thus, methods that assume Gaussian distribution in its dynamic motion model such as PF fail in such cases. The changes between detection accuracy on complete and sampled sequence is as indicated in red in Fig. 20. SwATrack on the other hand, copes better with low frame rate with an average accuracy of more than 70% although there is a decrease in its efficiency. This is because, the proposed SwATrack algorithm allows iterative adjustment of the exploration and exploitation of the swarm in search for the optimal motion model without making assumptions on the object's motion. We can thus conclude that the proposed SwATrack algorithm is able to cope with scenarios where the frame rate is low.

7. Conclusions

In this paper, we presented a novel swarm intelligence-based tracker for visual tracking that copes with abrupt motion efficiently. The proposed SwATrack optimised the search for the optimal distribution without making assumptions or need to learn the motion model before-hand. In addition, we introduced an adaptive mechanism that detects and responds to changes in the search environment to allow on the tuning of the parameters for a more accurate and effective tracking. To the best of our knowledge, this has never been done before. A new dataset – the Malaya Abrupt Motion *(MAMo)* dataset that consists of 12 videos with groundtruth is also provided. Experimental results show that the proposed algorithm improves the accuracy of tracking while significantly reduces the computational overheads, since it requires less than 20% of the samples used by PF. In future, we would like to further investigate the robustness of the proposed method as well as its behaviour change with the different parameter settings and sampling strategy.

8. Acknowledgement

This research work was supported by the University of Malaya HIR under Grant UM.C/625/1/HIR/037, J0000073579; and Mei Kuan Lim is sponsored by the Yayasan Khazanah Malaysia.

References

[1] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: CVPR, 2006, pp. 798–805.
[2] M.S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking, IEEE Trans. Signal Process. 50 (2) (2002) 174–188.
[3] C.S. Chan, H. Liu, Fuzzy qualitative human motion analysis, IEEE Trans. Fuzzy Syst. 17 (4) (2009) 851–862.
[4] C.S. Chan, H. Liu, B. David, N. Kubota, A fuzzy qualitative approach to human motion recognition, in: FUZZ-IEEE, 2008, pp. 1242–1249.
[5] M. Clerc, J. Kennedy, The particle swarm – explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evolut. Comput. 6 (1) (2002) 58–73.
[6] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: MHS, 1995, pp. 39–43.
[7] L. Ellis, N. Dowson, J. Matas, R. Bowden, Linear regression and adaptive appearance models for fast simultaneous modelling and tracking, Int. J. Comp. Vis. 95 (2) (2011) 154–179.
[8] M.G. Epitropakis, V.P. Plagianakos, M.N. Vrahatis, Evolving cognitive and social experience in particle swarm optimization through differential evolution: a hybrid approach, Inform. Sci. 216 (2012) 50–92.
[9] M. Everingham, L. Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, Int. J. Comp. Vis. 88 (2010) 303–338.
[10] C. García Cifuentes, M. Sturzel, F. Jurie, G.J. Brostow, Motion models that only work sometimes, in: BMVC, 2012, pp. 1–12.
[11] M. Isard, A. Blake, Condensation – conditional density propagation for visual tracking, Int. J. Comp. Vis. 29 (1) (1998) 5–28.
[12] J. Kwon, K.M. Lee, Tracking of abrupt motion using wang-landau monte carlo estimation, in: ECCV, pp. 387–400.
[13] J. Kwon, K.M. Lee, Visual tracking decomposition, in: CVPR, 2010, pp. 1269–1276.
[14] J. Kwon, K.M. Lee, Wang-landau monte carlo-based tracking methods for abrupt motions, IEEE Trans. PAMI 35 (4) (2013) 1011–1024.
[15] W. Li, X. Zhang, W. Hu, Contour tracking with abrupt motion, in: ICIP, 2009, pp. 3593–3596.
[16] Y. Li, H. Ai, T. Yamashita, S. Lao, M. Kawade, Tracking in low frame rate video: a cascade particle filter with discriminative observers of different life spans, IEEE Trans. PAMI 30 (10) (2008) 1728–1740.
[17] M.K. Lim, C.S. Chan, D. Monekosso, P. Remagnino, Swatrack: A swarm intelligence-based abrupt motion tracker, in: Proceedings of IAPR MVA, 2013, pp. 37–40.
[18] H. Liu, F. Sun, Efficient visual tracking using particle filter with incremental likelihood calculation, Inform. Sci. 195 (2012) 141–153.
[19] Y. Liu, S. Lai, B. Wang, M. Zhang, W. Wang, Feature-driven motion model-based particle-filter tracking method with abrupt motion handling, Opt. Eng. 51 (4) (2012).
[20] E. Maggio, A. Cavallaro, Accurate appearance-based bayesian tracking for maneuvering targets, Comp. Vis. Image. Und. 113 (4) (2009) 544–555.
[21] F. Neri, E. Mininno, G. Iacca, Compact particle swarm optimization, Inform. Sci. 239 (2013) 96–121.
[22] M. Oussalah, J.D. Schutter, Possibilistic kalman filtering for radar 2d tracking, Inform. Sci. 130 (14) (2000) 85–107.
[23] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: WCCI, 1998, pp. 69–73.
[24] K. Smith, D. Gatica-Perez, J. Odobez, S. Ba, Evaluating multi-object tracking, in: CVPRW, 2005, pp. 36–36.
[25] C. Sun, J. Zeng, J. Pan, S. Xue, Y. Jin, A new fitness estimation strategy for particle swarm optimization, Inform. Sci. 221 (2013) 355–370.
[26] G. Tong, Z. Fang, X. Xu, A particle swarm optimized particle filter for nonlinear system state estimation, in: CEC, 2006, pp. 438–442.
[27] A. Torralba, A.A. Efros, Unbiased look at dataset bias, in: CVPR, 2011, pp. 1521–1528.
[28] F. van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, Inform. Sci. 176 (8) (2006) 937–971.
[29] E.A. Wan, R. Van Der Merwe, The unscented kalman filter for nonlinear estimation, in: AS-SPCC, 2000, pp. 153–158.
[30] F. Wang, M. Lu, Hamiltonian monte carlo estimator for abrupt motion tracking, in: ICPR, IEEE, 2012, pp. 3066–3069.
[31] H. Wang, H. Sun, C. Li, S. Rahnamayan, J.-S. Pan, Diversity enhanced particle swarm optimization with neighborhood search, Inform. Sci. 223 (2013) 119–135.
[32] G. Welch, G. Bishop, An introduction to the kalman filter, Tech. rep., Chapel Hill, NC, USA, 1995.
[33] K.C.P. Wong, L.S. Dooley, Tracking table tennis balls in real match scenes for umpiring applications, BJMCS 1 (4) (2011) 228–241.
[34] Y. Xia, Z. Deng, L. Li, X. Geng, A new continuous-discrete particle filter for continuous-discrete nonlinear systems, Inform. Sci. 242 (0) (2013) 64–75.
[35] F. Yan, W. Christmas, J. Kittler, A tennis ball tracking algorithm for automatic annotation of tennis match, Sig. Proc. (2005) 619–628.
[36] H. Yang, L. Shao, F. Zheng, L. Wang, Z. Song, Recent advances and trends in visual tracking: a review, Neurocomputing 74 (2011) 3823–3831.
[37] A. Yilmaz, O. Javed, M. Shah, Object tracking: a survey, ACM Comput. Surveys 38 (4) (2006) 13+.
[38] X. Zhang, W. Hu, S. Maybank, A smarter particle filter, in: ACCV, Springer, 2010, pp. 236–246.
[39] X. Zhang, W. Hu, S. Maybank, X. Li, M. Zhu, Sequential particle swarm optimization for visual tracking, in: CVPR, 2008, pp. 1–8.
[40] X. Zhang, W. Hu, X. Wang, Y. Kong, N. Xie, H. Wang, H. Ling, S. Maybank, A swarm intelligence based searching strategy for articulated 3d human body tracking, in: CVPRW, 2010, pp. 45–50.
[41] X. Zhou, Y. Lu, J. Lu, J. Zhou, Abrupt motion tracking via intensively adaptive markov-chain monte carlo sampling, IEEE Trans. Image Proc. 21 (2012) 789–801.
[42] X. Zhu, C. Vondrick, D. Ramanan, C. Fowlkes, Do we need more training data or better models for object detection?, in: BMVC, 2012, pp. 1–11.
[43] I. Zuriarrain, F. Lerasle, N. Arana, M. Devy, An mcmc-based particle filter for multiple person tracking, in: ICPR, 2008, pp. 1–4.