Jurnal Elektronika dan Telekomunikasi (JET), Vol. 25, No. 1, August 2025, pp. 28-37
Accredited by KEMDIKBUDRISTEK, Decree No: **158/E/KPT/2021**
doi: 10.55981/jet.688

Performance Comparison of Particle Filter, Optical Flow, and CSRT in Unsupervised Visual Tracking for Mobile Robots

Heru Taufiqurrohman[a,*], Abdul Muis[a], Yusuf Nur Wijayanto[b]
Tsani Hendro Nugroho[b], Zaid Cahya[b]

[a]Department of Electrical Engineering, Faculty of Engineering
Universitas Indonesia
Kampus UI Depok 16424
Depok, Indonesia
[b]Research Center for Electronics
National Research and Innovation Agency (BRIN)
KST Samaun Samadikun , Jl. Sangkuriang, Dago, Coblong, 40135
Bandung, Indonesia

Abstract

This study addresses the challenges of selecting a suitable visual tracking method for real-time mobile robot applications, particularly in scenarios where the target is moving on the ground. The primary research problem addressed is the need for a flexible, computationally efficient tracking method that does not rely on pre-existing labeled datasets, as is often required by deep learning approaches. Unsupervised methods can overcome this problem by utilizing object motion information in each image frame without prior training. With many unsupervised tracking methods available, choosing an appropriate algorithm that can perform efficiently under dynamic conditions becomes a critical problem. The study compares the performance of three unsupervised visual tracking methods: particle filter, optical flow, and channel and spatial reliability tracker (CSRT) under various tracking conditions. The dataset used includes challenges such as moving target variations, changes in object scale, viewpoint changes, suboptimal lighting, image blurring, partial occlusions, and abrupt movements. Evaluation criteria include tracking accuracy, resistance to occlusion, and computational efficiency. The particle filter with ORB and a constant velocity model achieves a root mean square error (RMSE) of 36.47 pixels at 13 frames per second (fps). Optical flow performs best with an RMSE of 10.79 pixels at 30 fps, while CSRT shows an RMSE of 252.35 pixels at 4 fps. These findings highlight the effectiveness of optical flow for real-time applications, making it a promising solution for mobile robot visual tracking in challenging situations.

Keywords: Unsupervised visual tracking, mobile robots, particle filter, optical flow, CSRT, real-time tracking.

I. Introduction

Visual tracking is critical in various fields, such as autonomous robotics, surveillance, and video analysis. Recent advancements in deep learning-based methodologies have significantly enhanced tracking accuracy. One of the main problems is the limitation of deep learning-based methods, which require a lot of time and cost for the training process. In addition, this approach also requires a large labelled dataset, which is not always available, especially for real-time applications [1], [2]. Another challenge is the need for a flexible object-tracking method that can adapt to changing environmental conditions without relying on a pre-trained dataset [3]– [5]. This method is important because real-world tracking targets are often new or unknown objects.

Along with the development of technology, unsupervised tracking methods have become an attractive alternative. Unsupervised methods can overcome this problem by utilizing object motion information in each image frame without prior training. With the many unsupervised methods available, selecting a visual tracking method for mobile robots with targets moving on the ground becomes a problem when we implement it in a real-time system. Therefore, this study evaluates unsupervised visual tracking methods, which are more suitable for application in such conditions.

This study compares three popular unsupervised tracking methods: particle filter, optical flow, and channel and spatial reliability tracker (CSRT). These three methods represent visual tracking algorithms with different characteristics in terms of speed, robustness to occlusion, and adaptability to target changes. The following are the characteristics of each method. Particle filter is suitable for use in environments with non-linear motion and can handle object shape and position changes. However, this method has a high computational burden, which can hinder its performance on devices with limited resources [6], [7].

Particle filters are sophisticated algorithms frequently used for state estimation in various applications, especially navigation and robotics. They work well in dynamic contexts because they can handle non-linear and non-Gaussian issues. However, obstacles

* Corresponding Author.
Email: heru.taufiqurrohman31@ui.ac.id
Received: October 31, 2024 ; Revised : December 7, 2024
Accepted: December 23, 2024 ; Published: August 31, 2025

like initialization problems and particle deterioration may reduce its efficacy [8].

Studies on particle filters in robotics include enhanced particle filter performance for indoor navigation [9], improved detection probability for multi-target tracking [10], and improvements in navigation accuracy and robustness in mapping [11], the proposed method effectively addresses multiple solutions in inverse kinematics [12]. It also improves GPS positioning accuracy and robustness [13], and visual target locking during fast ground maneuvers [5]. The visual object tracking method that employs the particle filter algorithm, enhanced with Orientated Fast and Rotated BRIEF (ORB), provides a fast and accurate solution for object tracking [1]. The performance of the ORB particle filter (ORBPF) has been evaluated against traditional tracking methods under challenging conditions, including variations in illumination, scale, rotation, and occlusions [14].

Optical flow: this method is fast and efficient in calculating object motion between frames. Still, it is susceptible to noise and drastic changes in the target, such as changes in lighting or object shape [15]. Optical flow-based tracking methods offer a compelling alternative, particularly unsupervised techniques like the Lucas-Kanade algorithm. These methods focus on the motion patterns of pixels over time, enabling them to adapt swiftly to changing conditions without needing pre-trained models [16], [17]. Such characteristics make optical flow methods particularly suitable for real-world applications that demand speed and adaptability [18]. Recent research using optical flow includes detecting and tracking moving objects in video by analyzing velocity vectors [19], reconstructing global motion-group parameters from real-time image sequences to follow moving objects [20], detecting and tracking moving objects [21]– [23], vehicle motion tracking and velocity estimation [24].

CSRT: designed for high-accuracy object tracking, using a tight bounding box around the object. Although very accurate, this method has a low speed on devices with limited computing capacity [25]. Recent research using CSRT includes mobile object tracking [26]– [28], vision-guided manipulator operating system [29], and improved vehicle detection and tracking [30].

To comprehensively understand visual tracking capabilities, this research will compare three unsupervised visual tracking methods: Particle filter, optical flow, and CSRT. The focus on single-object tracking simplifies the scenario, allowing for a more detailed performance analysis of each method in a controlled setting [31]

In summary, even though deep learning techniques are renowned for their precision in visual tracking, their drawbacks in dynamic situations and real-time applications make other strategies necessary. This study compares particle filter, optical flow, and CSRT methods to determine the most suitable visual tracking method for implementation on a mobile robot with a moving target on the ground. It gives a better picture of what they can do in single-object tracking situations. This study also initiates further research into visual tracking systems using pan-tilt control cameras, broadening the tracking area.

II. METHODS

This research rigorously assessed the particle filter, optical flow, and CSRT algorithms for single-object tracking under controlled conditions for mobile robotics. Subsequently, the methodologies are elaborated, followed by an exposition of the experimental setup and performance evaluation metrics.

A. Particle Filter Algorithm

The sequential Monte Carlo approach, known as the color-based particle filter (CPF), works with any state model and is based on a representation of the mass of the point of the probability density [3]. The Monte Carlo algorithm is a sampling method to estimate a function's integral or numerical value, which is widely recognized in various fields [3]. The order of the data does not influence this method, as it relies on drawing random samples from a particular distribution to compute the estimate [3]. However, the Monte Carlo approach needs to consider time, as it typically considers only the distribution at that moment, reflecting its static nature [32]. The Particle Filter integrates sequential prediction, measurement, and resampling with aspects of Monte Carlo methods, allowing for a dynamic state estimate update [33]. Predicting system changes, taking fresh measurements, and updating the particles based on the measurements' level of agreement are all part of each time step in this framework [34].

Particle filtering is an advanced technique for estimating evolving states in systems defined by non-Gaussian distributions and non-linear dynamics. This approach is especially beneficial for tracking objects in complex and unpredictable environments, such as those found in robotics, navigation, and environmental monitoring [35]. The particle filter depicts the system's posterior distribution through a collection of particles, each representing a possible system state. Main steps of the particle filter algorithm:

1) Particle Initialization:

The preliminary step involves generating particles dispersed across a region where the target is likely to be located. This configuration is crucial as it establishes the foundation for subsequent estimation processes. The particles are usually sampled from a prior distribution, utilizing available information regarding the target's location, which may be informed by historical observations or an environmental model [8], [36].

2) Prediction

During this phase, every particle progresses according to a dynamic model that characterizes the target's movement. This model can include various elements such as speed, acceleration, and environmental factors, allowing the particles to adjust over time. The prediction phase is crucial to preserve the particle's importance as the system dynamics shift. Recent research has indicated that incorporating Kalman filtering techniques can improve prediction precision, particularly in occlusion situations [36].

3) *Weighting*

After the prediction step, the process necessitates determining the weight of each particle by evaluating its congruence with the current observation, such as the observed position of an object in an image or video. This procedure typically involves a likelihood function which quantifies the probability of the current data given the state of the particle. Particles that closely align with the observations are assigned higher weights, whereas those with fewer congruent matches are assigned lower weights. This step is crucial to focus the filter on the most probable states of the target [37].

The fundamental formula for calculating the weight w_i of particle *i* at time *t* is based on the current state (x_t) and the current observation (z_t). The weight is defined as:

$$w_i^t = P(z_t|x_t^i) \tag{1}$$

Here, $P(z_t|x_t^i)$ represents the likelihood of receiving observation z_t given that the particle is in state x_t^i.

4) *Resampling*

The final phase involves the resampling of particles. During this process, particles with low weights are discarded, and new particles are generated near those with high weights. This approach facilitates the focusing of computational resources on the most promising regions of the state space. It mitigates the issue of particle degeneracy, wherein a small number of particles dominate the representation of the posterior distribution. Advanced resampling strategies, such as systematic or stratified resampling, can enhance the effectiveness and precision of the particle filter [38].

Selecting a target motion model significantly impacts the algorithm's performance and precision within visual tracking algorithms that utilize particle filters. A commonly utilized model in this regard is the constant velocity (CV) model, which assumes the target retains a consistent speed and trajectory over time [39]. Object tracking model with CV model with linear discrete state equation:

$$x_{k+1} = Fx_k + v_k \tag{2}$$

where x_{k+1} is the tracker employs a state vector that includes the target's position (x, y), the velocity ($\dot{x}$, $\dot{y}$) and the scale s. The noise within the object is denoted by v_k. x_{k+1}, x_k, and v_k are in pixels. ^{T} is the transpose symbol, indicating that this is a column vector.

$$x_k = [x_k\ \ y_k\ \ \dot{x}_k\ \ \dot{y}_k\ \ s\ ]^T \tag{3}$$

and F represents the transition matrices:

$$\mathrm{F} = \begin{bmatrix} 1 & 0 & D & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & D & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

where D is a constant parameter obtained from the covariance of the process matrices about the detected object. Given that D is presumed constant, the velocity of the tracked object is consequently constrained.

The constant velocity model provides a straightforward approach to predicting a target's future position by analyzing its current state. This model assumes that the velocity remains unchanged, making it appropriate for various practical situations where targets exhibit consistent motion [12]. However, the model may face challenges when the target experiences abrupt changes in speed or direction, such as during evasive maneuvers or navigating complex environments [6].

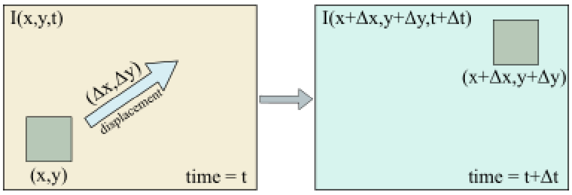

Figure 1. Basic concept of optical flow.

The sophisticated motion model incorporates constant velocity constant turn (CVCT), wherein ω is the turning rate.

$$x_k = [x_k\ \ y_k\ \ \dot{x}_k\ \ \dot{y}_k\ \ \omega\ \ s\ ]^T \tag{5}$$

and the transition matrices:

$$\mathrm{F} = \begin{bmatrix} 1 & 0 & \frac{\sin(\omega T)}{\omega} & -\frac{\cos(\omega T)}{\omega} & 0 & 0 \\ 0 & 1 & -\frac{\cos(\omega T)}{\omega} & \frac{\sin(\omega T)}{\omega} & 0 & 0 \\ 0 & 0 & \cos(\omega T) & -\sin(\omega T) & 0 & 0 \\ 0 & 0 & \sin(\omega T) & \cos(\omega T) & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Previous research endeavored to employ the enhanced particle filter algorithm in conjunction with ORBPF for visual target tracking during rapid terrestrial movements, utilizing the CVCT target model. The standard ORBPF with CV method yielded a root mean square error (RMSE) proximal to 345,518 pixels, whereas the ORBPF-CVCT approach exhibited an RMSE of approximately 191,387 pixels. These outcomes suggest that the ORBPF-CVCT technique demonstrates superior proficiency in tracking rapidly maneuvering objects compared to the ORBPF-CV method [5].

B. Optical Flow Algorithm

The concept of optical flow is based on the assumption that the pattern of pixel intensities of an object in an image does not change as the object moves. If an object moves from one position to another in a sequence of images, the intensity of the pixels representing that object remains constant, even though its position changes [40]. This is illustrated in Figure 1.

The Lucas-Kanade method was chosen for its efficiency and suitability for real-time applications on mobile robots. This method calculates optical flow by solving the motion equations locally for each pixel in a small neighborhood, making it computationally lighter than dense methods like Farneback. It is particularly effective in environments with good structure and moderate motion but may face challenges with rapid movements or large displacements.

Mathematically, optical flow can be represented by the intensity continuity equation, which is stated as follows:

$$I(x,y,t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{7}$$

where *I(x,y,t)* is the pixel intensity at coordinate (*x,y*) at time *t*, *Δx* and *Δy* are the changes in position in the *x* and *y* directions over time, and *Δt* is the change in time. By expanding the equation using a Taylor series and neglecting higher terms, we obtain the following equation:

$$\frac{\delta I}{\delta x}u + \frac{\delta I}{\delta y}v + \frac{\delta I}{\delta t} = 0 \tag{8}$$

δI/δx and *δI/δy* are the image intensity gradients in the *x* and y directions, *δI/δt* is the image intensity gradient in time, and *u* and *v* are the optical flow velocity components in the *x* and *y* directions.

C. CSRT

The CSRT represents a robust correlation-based object-tracking methodology. It exploits both chromatic (channel) and spatial information to augment the tracking precision.[25]. This methodology demonstrates notable efficacy in addressing scale and rotational dynamics variations, rendering it highly applicable to a diverse spectrum of tracking applications. [25], [41]. The main procedural steps include:

1) *Feature Extraction*

Initially, the CSRT algorithm employs histogram features or features derived from the Histogram of Oriented Gradients (HOG) to encapsulate the characteristics of the object. These features are crucial for distinguishing the target object from the background and other entities in the scene. The selection of suitable features markedly influences the robustness of the tracking process, especially in challenging scenarios such as occlusions or variations in lighting [42].

2) *Model Update*

After extracting features, the tracking model undergoes an update wherein information about the object derived from the current frame is incorporated. This updating process facilitates the model's adaptation to any modifications in the object's appearance or positioning, thus preserving tracking accuracy over time. The updating of the model is critical for addressing variations in scale, rotation, and other transformations that the object may experience during the tracking process [25].

3) *Matching*

The next step involves applying correlation filtering to locate the object in the subsequent frame. This process uses the updated model to search for the object based on the extracted features. The correlation filter computes the similarity between the current frame and the model, allowing for precise object localization. This matching step is critical for ensuring that the algorithm can accurately track the object despite potential changes in its appearance or position [25]. To estimate the object's position, CSRT applies correlation in the Fourier domain using the Fourier transform F :

$$y = F^{-1}\left(\frac{\sum_i F(f_i).\overline{F(f_i)}}{\sum_i F(f_i).F(f_i)}\right) \tag{9}$$

where f_i is the object's feature, the result of this correlation indicates the peak location, showing the object’s position.

D. Custom Dataset

The tracking algorithm will be tested with an image sequence dataset and compared with other well-known unsupervised detection algorithms. We created the dataset: video recordings of target movements modeled with a 1:10 scale RC car whose speed varies from 0 to 2 m/s. The video represents the challenges in tracking visual objects, including objects moving left, right, forward, and backward, then the challenges: changes in object scale to small or large, changes in front, side, and back views, poor lighting, blurred images, loss of part of the object, sudden movement, and changing speed. The video was taken with a still camera position with 30 fps and a resolution of 640 × 480 px. Then, the video was made into an image sequence of 2090 images, and an example is shown in Figure 2.

Annotation of the image sequence dataset was carried out using Visual Object Tagging Tool (VoTT) software, namely, providing annotations in the form of bounding boxes on the target image, as shown in Figure 3. The annotation results for each frame obtained ground truth in the form of four points of the x and y positions of the box forming the bounding box against the image frame, namely x_{min}, y_{min}, x_{max}, and y_{max} so that the position (*x*, *y*) of the center point can be calculated against the frame.

E. Experimental Setup

We used a mobile robot scale 1:10 equipped with a camera, a jetson nano dev kit mini-computer, and a 5-volt DC power supply to provide power to the mini PC and camera, as shown in Figure 4, and the target we used the RC car scale 1:10 as shown in Figure 5.

Target lock testing on all algorithms was conducted using a Jetson nano with ARM Cortex A57 running Ubuntu 20.04 with 4 GB of memory and 1.43 GHz of processing power. The data sets were, in turn, tested to determine the actual computing speed when applied to this hardware. This test is carried out by running the program, which selects the target or region of interest (ROI) manually and then runs the tracker algorithm to track the target from the custom-created image sequence. The test results in the form of the tracked targets' x and y center point positions will be compared with the ground truth in each frame by calculating the root mean square error (RMSE) of the center point. The computational

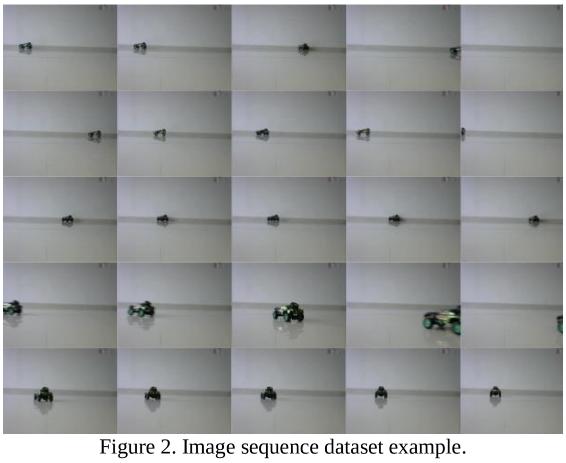

Figure 2. Image sequence dataset example.

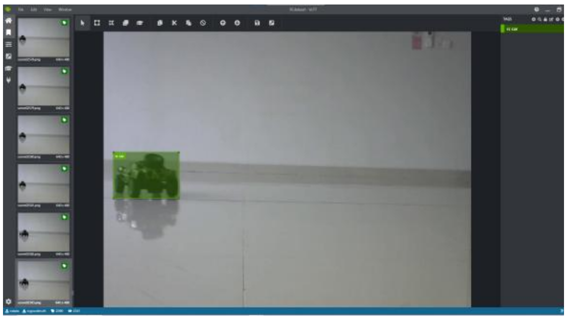

Figure 3. Image sequence annotation using VoTT software.

speed is also recorded. From these results, an analysis is performed so that the performance of each algorithm can be determined.

This experiment uses a visual tracking particle filter algorithm, incorporating a particle filter integrated into ORB with CV and CVCT motion models. The methodology commences with an initialization phase, where the target for tracking is selected by delineating an ROI in the form of a bounding box. After that, the ORB and PF algorithms operate simultaneously. The ORB algorithm identifies features by pinpointing critical points using FAST, and BRIEF calculates descriptors for each detected feature. Concurrently, the particle filter algorithm generates many particles and disperses them within the ROI.

After this, weights are allocated to the critical points ORB identifies and the particles. The amalgamation of these significant points produces points with enhanced prediction probabilities. The assignment of weights to the particles is predicated upon their respective potential weights. Particles exhibiting higher likelihoods are regarded as more probable and hence are assigned greater weights, whereas particles with lower probabilities are accorded lesser weights. Afterward, particles are arranged in descending order based on their weights, and those with optimal weights are selected for future application.

The next step involves normalizing and resampling the weights to create a new set of particles for future iterations. Finally, the following stage includes computing and estimating new positions using a particular dynamic model.

The optical flow tracker algorithm process begins with initialization and preparation, in which crucial variables and objects, including video recordings and buffers for processed frames, are initialized. Additionally, signal handling is configured to allow the program to terminate safely if required. Subsequently, the user designates an ROI in the initial frame, which is a reference area for feature detection. Feature detection is subsequently initialized by utilizing the Shi-Tomasi corner detector to generate reliable tracking features. Moving to frame processing, each new frame is read from the input source, comprising a series of images from a directory. Frames are converted to grayscale to enhance the efficiency of feature detection. Background subtraction is applied to produce a foreground mask, thus isolating moving objects from the static background. Optical flow is computed using the Lucas-Kanade method to track the movement of features between frames. This method employs pyramid images to capture changes in the positions of feature points. Points are filtered according to the status returned by the Lucas-Kanade method, ensuring that only successfully tracked points are considered. Should the number of tracked features satisfy the required threshold, the Random Sample Consensus (RANSAC) method is utilized in conjunction with a homography search to compute the homography between the two matched point sets and exclude outliers. The resulting inliers are thus employed to define a new bounding box. The global and central bounding boxes are updated based on the tracked and filtered features, enabling the program to monitor the overall movement of an object within a video or image sequence.

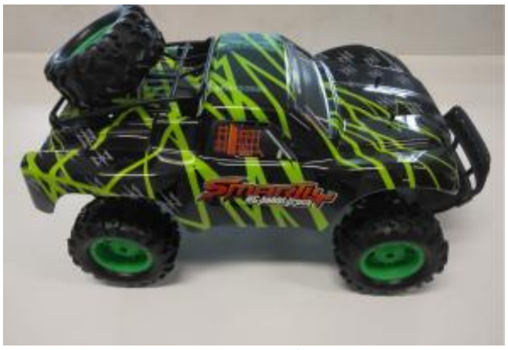

Figure 5. RC car 1:10 for the target.

The CSRT algorithm commences by selecting a ROI in the initial frame to serve as the reference area for object tracking. Following the definition of ROI, the CSRT tracker becomes operational. As a feature-based tracking method, CSRT demonstrates proficiency in tracking nonrigid objects. It employs features such as histogram of Oriented Gradients (HOG) and segmentation, which contribute to enhanced tracking stability even under varying conditions of shape or illumination. Each frame undergoes preprocessing through the normalization of the color intensity and applying Gaussian blur to mitigate noise, thereby facilitating feature detection. Within each frame, the object's bounding box position is updated dynamically. Upon successful tracking, the position of the bounding box is refined using a moving average technique that calculates the average of bounding box positions from preceding frames. This methodology mitigates jitter and improves tracking stability.

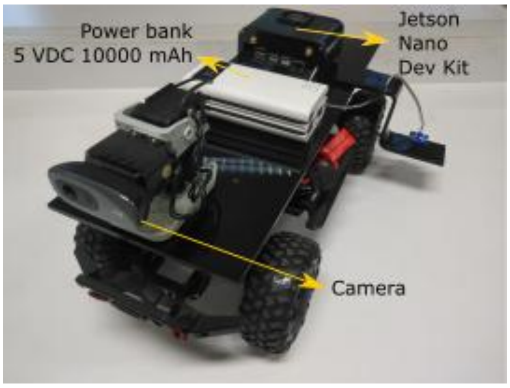

Figure 4. Mobile robot configuration.

Ultimately, the updated bounding box and tracking outcomes are rendered in a live output window, ensuring that object tracking is exhibited in real-time with stability and reliability.

All visual tracking algorithms have been developed by utilizing several opencv libraries, with the main focus of evaluation being as follows:

1) Accurate tracking

Evaluated by comparing the program's effectiveness in tracking the target object across different scenarios.

2) Efficiency of Computation

The cost-effectiveness of utilizing frames per second (fps) for real-time applications was assessed.

3) Strength

It was tested under challenging conditions, such as moving left, right, forward, and backward, scaling objects to small or large, front, side, and back views, dark conditions, blur, partially disappearing objects, abrupt movements, and speed changes.

III. Result and Discussion

This study evaluates the effectiveness of various unsupervised visual tracking methods on mobile robots, including ORBPF-CV and ORBPF-CVCT particle filters, optical flow, and CSRT. We use a dataset that has been annotated with ground-truth target center points. We assess its accuracy by measuring the error of the dataset's center point with the tracking result's center point in RMSE. We measure the computational speed of each algorithm. Figure 6 shows an example of ORBPF tracking results where ORBPF with the target dynamic model using CV and CVCT is performed simultaneously, where the green bounding box is the ORBP-CV, and the yellow bounding box is the ORBPF-CVCT. Figure 7 and 8 are examples of optical flow tracking and CSRT tracking results.

In the initial experiment, a comparative analysis was conducted between the ORBPF-CV algorithms under three distinct scenarios, each characterized by a specific combination of particles in the particle filter (PF) and the ORB. The ratios examined include 50:20, 80:30, and 100:40. This investigation aimed to assess the balance between accuracy and computational efficiency.

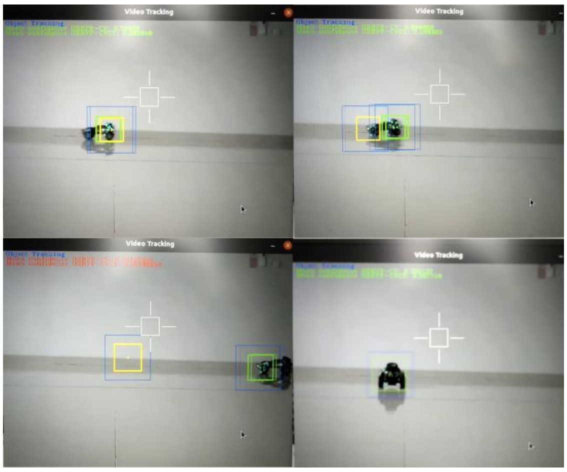

Figure 6. Example of ORBPF tracking results.

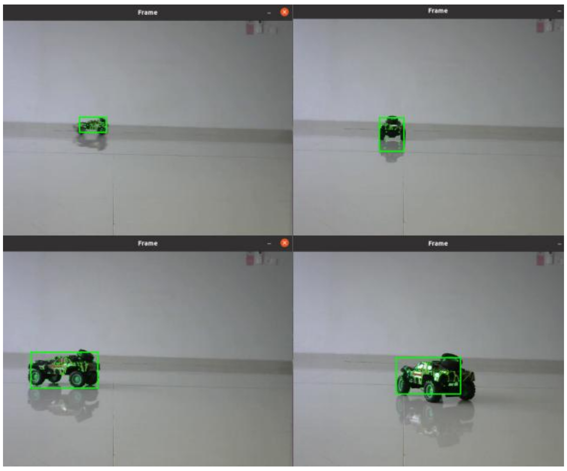

Figure 7. Example of optical flow tracking results.

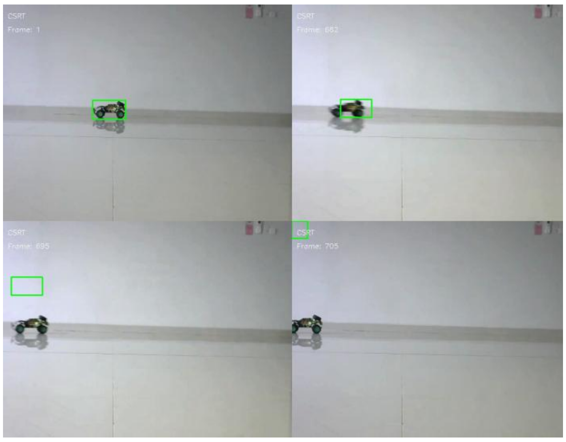

Figure 8. Example of CSRT tracking results.

Figure 9 illustrates the results for a scenario characterized by a ratio of 50:20, where the mean processing duration per frame of ORBPF-CV, corresponding to 13 fps, is 0.076263 seconds with RMSE of 36.47 pixels. On the contrary, ORBPF-CVCT records a processing time of 0.079559 seconds or 12 fps, with an RMSE of 86.26 pixels. Figure 10 presents the results for a scenario with a ratio of 80:30, indicating that the average processing time per frame of ORBPF-CV is 0.106789 seconds or 9 fps, with an RMSE of 33.84 pixels. In comparison, ORBPF-CVCT shows a

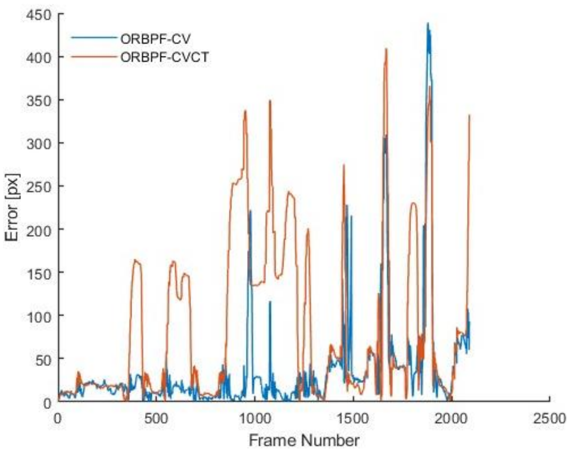

Figure 9. Test results with a combination of PF particles and ORB particles of 50:20.

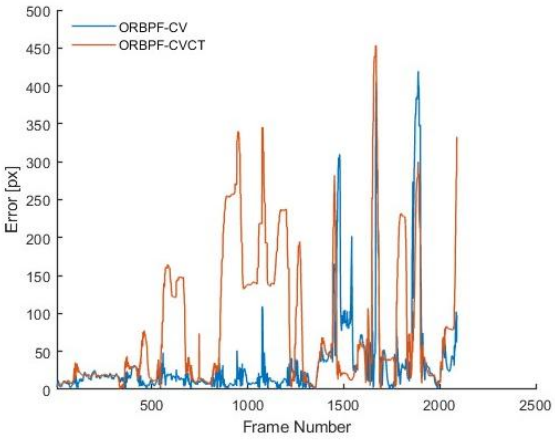

Figure 10. Test results with a combination of PF particles and ORBparticles 80:30.

processing time of 0.111047 seconds or 9 fps, with an RMSE of 82.42 pixels. Figure 11 shows the results for a scenario with a ratio of 100:40, revealing that the average processing time per frame of ORBPF-CV is 0.148283 seconds or 7 fps, with an RMSE of 25.61 pixels. In comparison, ORBPF-CVCT demonstrates a processing time of 0.1554036 seconds or 6 fps, with an RMSE of 98.14 pixels.

This analysis indicates that increasing the ratio of particles in the particle filter (PF) and ORB progressively reduces the RMSE for ORBPF-CV. In contrast, it tends to elevate the RMSE for ORBPF-CVCT, coupled with a decrease in computational speed for both methods.

When ORBPF-CV is compared with ORBPF-CVCT in this specific challenge, ORBPF-CV exhibits superior performance. Both algorithms exhibit comparable characteristics in their RMSE peaks, particularly when the target undergoes abrupt movements and experiences partial or complete loss. However, ORBPF-CV demonstrates a more expedient capability to reacquire the target.

The constant velocity model is considered superior for tracking ground RC targets within this scenario due to its simplicity and enhanced responsiveness to changes in linear velocity, devoid of any rotational elements. It proves appropriate for linear motion patterns that lack noticeable turns, facilitating rapid and precise predictions when the target experiences abrupt stops or accelerations. On the contrary, the constant turn model introduces superfluous complexity, which may increase the likelihood of predictive inaccuracies during sudden forward or backward movements.

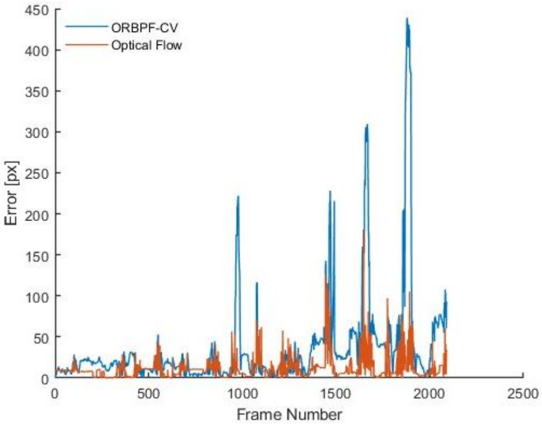

Figure 12. Test results ORBPF-CV vs. Optical Flow.

Particle filter has the advantage of handling complex motion and non-Gaussian noise. However, this algorithm is susceptible to drift problems, especially when the target has an irregular shape or is partially occluded. In our tests, drift occurs when the target loses its main features (key points) due to partial loss of the target shape or significant sudden target movement, which causes the algorithm to learn background features because the particle filter relies on local feature models (such as ORB), which are susceptible to variations in the target shape [43]. The tracking results show a decrease in accuracy with an RMSE of up to 36.47 pixels in such situations.

The subsequent phase involves evaluating the optical flow algorithm. The findings reveal an average processing duration per frame of 0.033718526 seconds, approximately equivalent to a frame rate of 30 fps, with an accuracy measured by RMSE at 10.79 pixels. Compared to the optimal outcomes of ORBPF-CV, the performance of this optical flow algorithm demonstrates significant superiority. The results are illustrated in Figure 12. Both algorithms exhibit similar RMSE peaks,

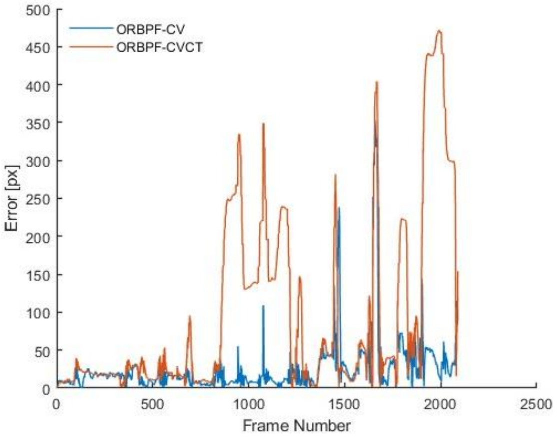

Figure 11. Test results with a combination of PF particles and ORB particles 100:40.

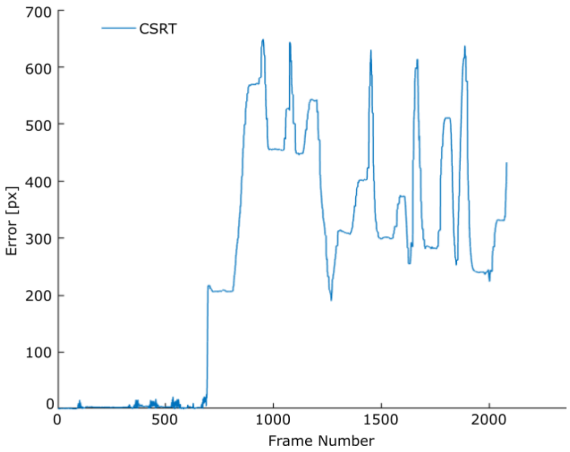

Figure 13. Test results of CSRT.

particularly when the target undergoes abrupt movements and encounters partial or complete occlusion. The subsequent experiment evaluated the CSRT algorithm to assess its performance.

The findings in Figure 13 indicate a processing time per frame of 0.252350329 seconds, equivalent to approximately 4 fps, with an RMSE of 252.35 px. This constitutes the least effective performance among all algorithms for the target movement scenarios represented in this dataset. The tracking algorithm cannot maintain a lock on the target when it initiates rapid movement and exits the frame. This limitation arises because of the relatively prolonged processing time per frame, which is inadequate for a real-time system, rendering it highly vulnerable to losing the target under swift movement and frame disappearance conditions.

CSRT performance is highly dependent on the quality of the spatial reliability map, which is affected by the accuracy of colour and texture segmentation. In our tests, CSRT showed degraded performance under conditions such as poor lighting, blurred images, and significant changes in target scale [44].

In low-light conditions, colour segmentation of the spatial reliability map becomes less accurate, leading to tracking errors. CSRT has difficulty maintaining tracking on blurred images, resulting in an RMSE of up to 252.35 pixels. This result suggests that CSRT is not ideal for highly variable real-world conditions unless data preprocessing, such as image enhancement, is performed before encryption [45]–[47].

CSRT uses channel and spatial reliability for more precise tracking, but this approach increases the computational complexity compared to algorithms such as optical flow or particle filter [48]. In this study, CSRT showed an average processing speed of 4 fps, much lower than optical flow (30 fps) and particle filter (13 fps). Although slow speed, CSRT provides fairly precise tracking results when the target changes in scale and shape. The low processing speed makes it less suitable for real-time applications in systems with limited computing power, such as mobile robots that require fast response [49].

Comparison of the test results of the visual particle filter tracking method, optical flow, and CSRT are summarized in Table 1, with particle filter data taken at its best performance. We evaluate the strength of each algorithm by measuring the accuracy or error of tracking results on test video frames that simulate real conditions, though with certain limitations. These conditions include changes in scale, shape transformations, sudden movements, poor lighting, and blurry images. This approach enables us to assess each algorithm's capability to handle dynamic and complex visual challenges. The test result graphs for each method display these findings, highlighting their RMSE peak points. The peak of the RMSE point indicates that the tracker algorithm has lost its target, and after the peak of the RMSE, the tracker algorithm tries to get its target back. The graphs show that optical flow demonstrates the strongest ability to handle challenges, followed by particle filters and CSRT.

TABLE 1
VISUAL TRACKING TEST RESULTS USING PARTICLE FILTERS, OPTICAL FLOW, AND CSRT

Method	RMSE (pixel)	Computational Speed (fps)	Strength Against Occlusion
Particle Filter	36,7	13	Moderate
Optical Flow	10,79	30	High
CSRT	252,35	4	Low

IV. CONCLUSION

This study demonstrates that optical flow-based unsupervised visual tracking, particularly employing the Lucas-Kanade algorithm, constitutes a highly efficacious approach for mobile robots in real-time applications. Among the three methods evaluated, optical flow performs best with a root mean square error (RMSE) of 10.79 pixels at a computational speed of 30 fps, showing high strength against occlusion. This performance surpasses particle filter-based tracking, which achieves an RMSE of 36.7 pixels at 13 fps with moderate strength against occlusion, and CSRT, which has the poorest performance with an RMSE of 252.35 pixels at 4 fps and low occlusion resistance. The study's findings show that optical flow works well in regulated and dynamic settings, making it a good option for real-time reaction applications, including target locking, self-navigating, and dynamic video capturing. Future research may explore hybrid methodologies that combine the strengths of optical flow with learning-based models to enhance robustness in complex environments. For example, combining deep learning techniques with the Lucas-Kanade method may enhance tracking effectiveness while dealing with sudden target motions and situations when targets vanish from the screen.

Furthermore, adding further elements like texture or color information may improve tracking accuracy even more in difficult situations. The techniques become more useful in practical situations by addressing issues like computing efficiency and data dependency. While model pruning and quantization can minimize computational burdens for real-time applications, transfer learning and few-shot learning techniques can lessen dependency on huge labeled datasets. Furthermore, online learning or reinforcement learning may enable models to adapt dynamically to new data in real-time, improving their effectiveness in dynamic and unpredictable environments. Another solution is integrating traditional methods with CNN-based methods, which might be a good way to provide more flexibility and tracking accuracy, especially in dynamic and complex situations.

DECLARATIONS

Conflict of Interest

The authors have declared that no competing interests exist.

CRediT Authorship Contribution

Heru Taufiqurrohman: Conceptualization, Methodology, Software, Writing-Original draf, Data Curation; Abdul Muis: Conceptualization, Methodology, Supervision, Writing-Review and Editing; Yusuf Nur Wijayanto: Conceptualization, Methodology, Supervision, Writing-Review and Editing; Tsani Hendro Nugroho: Methodology, Software, Data Curation; Zaid

Cahya: Conceptualization, Software, Software, Writing-Review and Editing.

Funding

The research received support from BRIN's Research Program on Autonomous Vehicles from the Research Organization of Electronics and Informatics, and the BRIN Degree by Research Program.

Acknowledgment

The author expresses gratitude for the support offered by the National Research and Innovation Agency (BRIN) and Universitas Indonesia.

REFERENCES

[1] B. Li, C. Fu, F. Ding, J. Ye, and F. Lin, "All-day object tracking for unmanned aerial vehicle."*IEEE Trans. Mob. Comp.*, vol. 22, no. 8, pp. 4515-4529, Aug, 2023, doi: 10.1109/TMC.2022.3162892.

[2] V. Devyatkov and I. I. Lychkov, "Semi-automatic multi-camera video annotation tool for object tracking and optical flow benchmark creation." in *Proceedings of the International Conferences on Interfaces and Human Computer Interaction 2019: Game and Entertainment Technologies 2019; and Computer Graphics, Visualization, Computer Vision and Image Processing 2019*, pp. 427-431, 2019. doi: 10.33965/cgv2019_201906c062.

[3] G. Wang, Y. Yang, and K. He, "A robust facial feature tracking method based on optical flow and prior measurement," *IJCINI*. 2010, pp. 62-75. doi: 10.4018/jcini.2010100105.

[4] K. Song, C. Yuan, P. Gao, and Y. Sun, "FPGA-based acceleration system for visual tracking."in *Proceeding - 2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology, ICSICT 2018,* 2018. pp. 993-995. doi: 10.1109/icsict.2018.8565781.

[5] H. Taufiqurrohman, A. Muis, Y. N. Wijayanto, T. H. Nugroho, D. E. Cahya, and Z. Cahya, "Visual target locking during fast ground maneuver using enhanced orb predictive particle filter," in *Proceeding - 2023 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications: Empowering Global Progress: Innovative Electronic and Telecommunication Solutions for a Sustainable Future, ICRAMET 2023*, 2023, pp. 67–72. doi: 10.1109/ICRAMET60171.2023.10366613.

[6] M. A. Awal, M. A. R. Refat, F. Naznin, and M. Z. Islam, "A particle filter based visual object tracking: a systematic review of current trends and research challenges," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 14, no. 11, pp. 1291-1301, 2023. doi: 10.14569/ijacsa.2023.01411131.

[7] J. Wang, D. Zhao, S. Shan, and W. Gao, "Approximating inference on complex motion models using multi-model particle filter." LNCS. *Advances in Multimedia Information Processing - PCM 2004, vol.3332*, pp. 1011-1018, 2004. doi: 10.1007/978-3-540-30542-2_124.

[8] J. Majumdar, P. Dhakal, N. S. Rijal, A. M. Aryal, and N. K. Mishra, "Implementation of hybrid model of particle filter and kalman filter based real-time tracking for handling occlusion on beagleboard-xm," *International Journal of Computer Applications*, vol. 95, no.7, 2014, pp. 31-34. doi: 10.5120/16608-6443.

[9] R. Giri, A. Kumar Sah, and S. Satyal, "Particle filter enhanced by cnn-lstm for localization of autonomous robot in indoor environment," *Int. J. Engin. Technol.*, vol. 1, no. 2, pp. 1–12, Jun. 2024. doi:10.3126/injet.v1i2.66654.

[10] Y. Cheng, W. Ren, C. Xiu, and Y. Li, "Improved particle filter algorithm for multi-target detection and tracking," *Sensors (Basel)*, vol. 24, no. 14, p. 4708, Jul. 2024. doi: 10.3390/s24144708.

[11] Z. Rozsypalek, T. Roucek, T. Vintr, and T. Krajnik, "Multidimensional particle filter for long-term visual teach and repeat in changing environments," *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 1951–1958, Apr. 2023. doi: 10.1109/LRA.2023.3244418.

[12] C.-L. Chiang, C.-C. Hsieh, Y.-Y. Chiang, and I.-L. Lin, "Using particle filters to solve the problem of symmetric multiple solutions in robot inverse kinematics," *J. Internet Technol.*, vol. 25, no. 4, pp. 551–559, Jul. 2024.

[13] F. Ye, Y. Xiang, C. Jiang, and T. Wang Jinrong and Wang, "Application of particle filter algorithm in satellite positioning and navigation system," *J. Phys. Conf. Ser.*, vol. 2724, no. 1, p. 12036, Mar. 2024. doi: 10.1088/1742-6596/2724/1/012036.

[14] T. H. Nugroho, F. Mangkusasmito, B. R. Trilaksono, T. Indriyanto, and L. Yulianti, "Enhancing color-based particle filter algorithm with orb feature for real-time video tracking," in *2018 International Conference on Signals and Systems (ICSigSys)*, 2018, pp. 53–58. doi: 10.1109/ICSIGSYS.2018.8373567.

[15] D. Kesrarat and V. Patanavijit, "Verification of video reconstruction using bilateral-reverse directional global based optical flow over non-gaussian noise," *International Journal of Simulation Systems Science & Technology*. 2019. doi: 10.5013/ijssst.a.19.03.04.

[16] Y. Cheng, C. Li-yun, and C. Luo, "Moving target tracking algorithm based on improved optical flow technology," *The Open Automation and Control Systems Journal*., vol. 7, pp. 1387-1392. Sept, 2015. doi: 10.2174/1874444301507011387.

[17] I. H. Choi, J. M. Pak, C. K. Ahn, S.-H. Lee, M. T. Lim, and M. K. Song, "Arbitration algorithm of fir filter and optical flow based on anfis for visual object tracking," *Measurement*. vol. 75, 338-353. Nov. 2015. doi: 10.1016/j.measurement.2015.07.020.

[18] H. Akolkar, S. H. Ieng, and R. Benosman, "Real-time high speed motion prediction using fast aperture-robust event-driven visual flow," *Ieee Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 104-118, Jul. 2020. doi: 10.1109/tpami.2020.3010468.

[19] A. Tarasov and M. Nikiforov, "Detection and tracking of moving objects optical flow based," in *2024 International Russian Automation Conference (RusAutoCon)*, Sep. 2024, pp. 121–126. doi: 10.1109/RusAutoCon61949.2024.10694055.

[20] S. Karpuzov, G. Petkov, S. Ilieva, A. Petkov, and S. Kalitzin, "Object tracking based on optical flow reconstruction of motion-group parameters," *Information (Basel)*, vol. 15, no. 6, p. 296, May 2024. doi: 10.3390/info15060296.

[21] M. Alipour Sormoli, M. Dianati, S. Mozaffari, and R. Woodman, "Optical flow based detection and tracking of moving objects for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 9, pp. 12578–12590, Sep. 2024. doi: 10.1109/TITS.2024.3382495.

[22] K. C. Hari, S. Shrestha, and M. Pokharel, "Video object motion tracking using dense optical flow techniques," in *2023 Eighth International Conference on Informatics and Computing (ICIC)*, Dec. 2023, pp. 1–4. doi: 10.1109/ICIC60109.2023.10381969.

[23] R. Ferreira, J. de Castro Ferreira, and A. José Ribeiro Neves, "Object tracking using adapted optical flow," Artificial Intelligence. IntechOpen, Aug. 17, 2022. doi: 10.5772/intechopen.102863.

[24] G. L, Adarsh, G. Vinayan, Gokuldath, Ponmalar, and A. S, "Lucas kanade based optical flow for vehicle motion tracking and velocity estimation," in *2023 International Conference on Control, Communication and Computing (ICCC)*, May 2023, pp. 1–6. doi: 10.1109/ICCC57789.2023.10165227.

[25] A. Lukežič, T. Vojíř, L. Čehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter tracker with channel and spatial reliability," *Int J Comput Vis*, vol. 126, no. 7, pp. 671–688, Jul. 2018, doi: 10.1007/s11263-017-1061-3.

[26] Y. Cao and A. A. Boguslavsky, "Software technology for mobile object tracking based on the fusion of csrt and sam algorithms," *in Proceedings of the 34th International Conference on Computer Graphics and Machine Vision, GRAPHICON 2024*, pp. 500-508. 2024, doi: 10.25206/978-5-8149-3873-2-2024-500-508.

[27] S. Ilkin, F. K. Gulagiz, and S. Akcakaya Merve and Sahin, "Embedded visual object tracking system based on csrt tracker," *in Proceeding - 2022 International Conference on*

Electronics, Information, and Communication, ICEIC 2022, 2022. pp. 275-278, doi: 10.1109/ICEIC54506.2022.9748840.

[28] X. Farhodov, O.-H. Kwon, S.-H. Kang Kyung Won and Lee, and K.-R. Kwon, "Faster RCNN detection based opencv csrt tracker using drone data," *in Proceeding* - 2019 International Conference on Information Science and Communications Technologies, ICISCT) *2019*, Nov. 2019, pp. 1-3, doi: 10.1109/ICISCT47635.2019.9012043.

[29] S. Bu, L. Yan, X. Gao, P. Zhao, and C. K. Lim, "Vision-guided manipulator operating system based on CSRT algorithm," *Int. J. Hydromechatronics*, vol. 5, no. 3, pp. 260-274, Aug. 2022. doi: https://doi.org/10.1504/IJHM.2022.125091.

[30] I. C. Amitha and N. K. Narayanan, "Improved vehicle detection and tracking using yolo and csrt," in *Communication and Intelligent Systems*, Singapore: Springer Singapore, 2021, pp. 435–446. doi: 10.1007/978-981-16-1089-9_35.

[31] P. N. Skonnikov and D. Trofimov, "Pupil visual tracking algorithms for automated static perimetry systems," *The International Archives of the Photogrammetry Remote Sensing and Spatial Information Sciences*. 2021. doi: 10.5194/isprs-archives-xliv-2-w1-2021-195-2021.

[32] Q. Wang, J. Li, M. Zhang, Yang, and Chenhui, "H-infinity filter based particle filter for maneuvering target tracking," *Prog. in Electromagn. Res. B.*, vol. 30, pp. 103-116, Jun. 2011, doi: 10.2528/pierb11031504.

[33] L. M. Murray, A. Lee, and P. Jacob, "Parallel resampling in the particle filter," *J. Comput. Graph. Stat.*, vol. 25, no. 3, pp. 789–805, Aug. 2016. doi: 10.1080/10618600.2015.1062015.

[34] A. H. Abolmasoumi, A. Farahani and L. Mili, "Robust Particle Filter Design With an Application to Power System State Estimation," in *IEEE Trans. Power Syst.*, vol. 39, no. 1, pp. 1810-1821, Jan. 2024, doi: 10.1109/TPWRS.2023.3263203.

[35] Y. Hou, Y. L. Zhao, T. Sun, and J. M. Di, "Research on kalman particle filter-based tracking algorithm," *Adv. Mat. Res.*, vol. 461, pp. 571–574, Feb. 2012. doi: 10.4028/www.scientific.net/amr.461.571.

[36] J. Majumdar, P. Dhakal, N. S. Rijal, A. M. Aryal, and N. K. Mishra, "Implementation of hybrid model of particle filter and kalman filter based real-time tracking for handling occlusion on beagleboard-xm," *Int. J. Comput. Appl.*, vol. 95, no. 7, pp. 31-37, Jun. 2014. doi: 10.5120/16608-6443.

[37] S. Yi, Z. He, X. You, and Y. Cheung, "Single object tracking via robust combination of particle filter and sparse representation," *Signal Processing*, vol. 110, pp. 178-187, May 2015. doi: 10.1016/j.sigpro.2014.09.020.

[38] Q. Meng and K. Li, "A rank correlation coefficient based particle filter to estimate parameters in non-linear models," *International Journal of Distributed Sensor Networks*, vol. 15, no. 4, April, 2019. doi: 10.1177/1550147719841273.

[39] M. Pupilli and A. Calway, "Real-time camera tracking using a particle filter." *in Proceedings of the British Machine Vision Conference 2005*. BMVC 2005. pp. 50.1-50.10, Sept. 2005. doi: 10.5244/c.19.50.

[40] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", in *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI) 1981*, vol.2, pp. 674-679, Aug. 1981. doi: 10.5555/1623264.1623280.

[41] J. Lussi *et al.*, "Magnetically guided laser surgery for the treatment of twin-to-twin transfusion syndrome," *Adv. Intell. Syst.*, vol. 4, pp. 1-12, Oct. 2022. doi: 10.1002/aisy.202200182.

[42] Q. Chen, H. Wang, and C. DONG, "Empirical analysis of vehicle tracking algorithms for extracting integral trajectories from consecutive videos," *Promet - Traffic&transportation*. PROMTT, vol. 34, no. 4, pp. 551–565, Jul. 2022. doi: 10.7307/ptt.v34i4.4041.

[43] Z. Liu, S. Wei, and J. C. Spall, "Error analysis for the particle filter." in *Proceeding - 2019 American Control Conference*. ACC. pp. 4515 – 4520, Aug. 2019. doi: 10.23919/ACC.2019.8814633.

[44] P. Shyam, A. Bangunharcana, and K.-S. Kim, "Retaining image feature matching performance under low light conditions." in *Proceeding - 2020 20th International Conference on Control, Automation and Systems*. ICCAS. pp. 1079-1085, Oct. 2020. doi: 10.23919/iccas50221.2020.9268426.

[45] H. Ackar, A. A. Almisreb, and M. A. Saleh, "A review on image enhancement techniques," *Southeast Europe Journal of Soft Computing*, vol. 8, no. 1, pp. 42-48, Mar. 2019. doi: 10.21533/scjournal.v8i1.175.

[46] P. Wang, G. Zhang, S. Hao, and L. Wang, "Improving remote sensing image super-resolution mapping based on the spatial attraction model by utilizing the pansharpening technique," *Remote Sensing*, vol. 11, no. 3, pp. 1-16, Jan. 2019. doi: 10.3390/rs11030247.

[47] D. A. Almoanf and S. H. Shaker, "Medical image enhancement techniques," *Iraqi Journal of Computer Communication Control and System Engineering*, vol. 22, no. 4, pp. 48-59, Dec. 2022. doi: 10.33103/uot.ijccce.22.4.5.

[48] K. Rapko, W. Xie, and A. Walsh, "MONCE tracking metrics: a comprehensive quantitative performance evaluation methodology for object tracking."Proc. SPIE 12096, *Automatic Target Recognition XXXII*. 12096-4, May 2022. doi: 10.1117/12.2618631

[49] J. Xie, E. Stensrud, and T. Skramstad, "Detection-based object tracking applied to remote ship inspection," *Sensors*, vol. 21, no. 3, pp. 1-23, Jan. 2021. doi: 10.3390/s21030761.