International Journal of Innovative Computing, Information and Control
Volume **18**, Number **3**, June **2022**

ICIC International ©2022 ISSN 1349-4198
pp. **723–738**

A FRAMEWORK FOR ROBUST OBJECT TRACKING BY COMBINING MULTIPLE OPENCV ALGORITHMS

Kohichi Ogata[1], Koki Tanaka[2], Rinka Ikeda[3] and Fitri Utaminingrum[4]

[1]Faculty of Advanced Science and Technology
[2]Graduate School of Science and Technology
[3]Faculty of Engineering
Kumamoto University
Kurokami 2-39-1, Chuo-ku, Kumamoto 860-8555, Japan
ogata@cs.kumamoto-u.ac.jp; tanaka.05.kk@gmail.com; c8107@st.cs.kumamoto-u.ac.jp

[4]Faculty of Computer Science
Brawijaya University
Jl. Veteran No. 8, East Java, Malang 65145, Indonesia
f3_ningrum@ub.ac.id

Received November 2021; revised March 2022

Abstract. *This paper describes a framework for robust object tracking by combining the results of several tracking algorithms. Object tracking is complicated by various factors, such as changes in the shape and/or movement direction of the tracked target, and occlusion by other objects. Object tracking algorithms each have individual strengths and weaknesses, and even an algorithm that is highly accurate in some circumstances may be less accurate in others. In the framework proposed here, the tracking results of several algorithms provided in OpenCV are integrated into a more reliable tracking result based on the centroid computed from a weighted average of center points determined by each individual algorithm. The results of experiments using video sequences demonstrate that the proposed framework was able to track a target object successfully even in cases in which most of the OpenCV algorithms individually failed to do so.*
Keywords: Robust object tracking, Detection, OpenCV, Algorithm, Computer vision

1. **Introduction.** Object tracking is an important topic in the field of computer vision. As surveillance cameras become more widely deployed and recorded images proliferate, the importance of related technologies is increasing in daily life. A wide variation in surveillance camera capabilities and related analyses and functionalities, including future trends, suggest the diversification of society's needs for object tracking technologies [1]. Various tracking methods have been proposed, and their accuracy has been improved [2-8]. A wide variety of tracking targets, such as cattle [9] and nutriments in fish feeding [10], suggest the expansion of applications of object tracking and expected future demand.

Object tracking is complicated by various factors, such as changes in the shape of the tracked object and the direction of movement. To overcome such issues, the development of improved tracking algorithms has continued. For example, an object tracking method based on a correlation filter with a saliency refiner and adaptive updating was proposed [11]. In this approach, visual saliency as prior information is integrated into a kernelized correlation filter to highlight salient objects and provide image features that are relatively invariant to appearance changes. Although this approach is effective, further improvement is necessary to meet practical performance expectations. Object tracking

DOI: 10.24507/ijicic.18.03.723

algorithms each have individual strengths and weaknesses, and even an algorithm that is highly accurate in some circumstances may be less accurate in others.

OpenCV (Open Source Computer Vision Library) [12] is an open-source computer vision and machine learning software library. Algorithms are available for a multitude of vision-related tasks including object tracking, such as KCF (Kernelized Correlation Filter) [5], MIL (Multiple Instance Learning) [6], Median Flow [7], and Boosting [8]. As mentioned above, object tracking algorithms have their own strengths and weaknesses depending on the scene type. Therefore, if object tracking depends only on a single algorithm, there is a limit to improving the tracking accuracy of the algorithm.

In this study, we developed a framework that allows us to effectively combine the tracking results of OpenCV algorithms. We expect that such a framework would benefit from combining the strengths of the individual algorithms, leading to more reliable tracking results.

Although there are limitations such as processing speed and computer resources required for parallel processing, the framework allows us to easily include existing algorithms in OpenCV and, therefore, obtain more reliable tracking results by taking advantage of the strengths of each algorithm. Even if the processing speed is unable to keep up with video in real time, the framework would still be helpful for the analysis of recorded video, such as log analysis of human and animal behavior. Our research motivation for the framework is based on these expected benefits. The contributions of this study are described in the Discussion, which details the benefits of the proposed framework.

The remainder of this paper is organized as follows. Section 2 presents a brief overview of the OpenCV object tracking algorithms used in this study. Section 3 proposes a framework for robust object tracking using multiple algorithms. The tracking results of several OpenCV object tracking algorithms are effectively combined to determine the object position more reliably. In Section 4, experiments are conducted to evaluate the usefulness of the framework for several video sequences. Section 5 discusses the benefits of the framework. Finally, the conclusions of this study are presented in Section 6.

2. **Related Work.**

In this section, a brief overview of the OpenCV object tracking algorithms used in the framework is presented. At the end of this section, our approach to robust object tracking is also described briefly to organize the motivation and expected contributions of the present work.

2.1. **KCF (Kernelized Correlation Filter) [5].**

In this method, an analytic model is used to generate datasets of thousands of translated patches. These datasets can be effectively produced by cyclic shifts of a base sample. Because the related resulting data matrix is circulant, the discrete Fourier transform effectively reduces storage and computation requirements. The Histogram of Oriented Gradient (HOG) [13] is used as a feature descriptor. This filter is a correlation filter. As an effective correlation filter, a tracker based on a Minimum Output Sum of Squared Error (MOSSE) filter was proposed and its robustness to variations in lighting, scale, pose, and nonrigid deformations was demonstrated in [14]. A preliminary version of KCF adapted from [14] was presented in [15], the results of which demonstrated the connection between ridge regression with cyclically shifted samples and classical correlation filters. The KCF [5] used HOG features instead of raw pixels in [15].

2.2. **MIL (Multiple Instance Learning) [6].**

A discriminative appearance model is trained and updated to separate an object from an image background via tracking-by-detection. Incorrectly labeled training examples degrade the performance of the classifier. To overcome this problem, a set of training examples called a "bag" is considered in this

method. During training, bags are prepared, and labels are provided for the bags rather than for individual instances. If a bag is labeled positive, it is assumed to contain at least one positive instance; otherwise the bag is negative. This method is based on a Multiple Instance Learning (MIL) [16] approach and is flexible in learning by allowing ambiguity of training examples.

2.3. **Median Flow [7].** In this method, reliable tracking is achieved based on the evaluation of forward-backward error, in which the discrepancies between these two directional trajectories are used as a measure of reliability. A bounding box, which surrounds an object, is determined for each image in a pair of images. Grid points are set within each bounding box, and these grid points are tracked by the Lucas-Kanade tracker [17, 18]. Grid points with reliable tracking trajectories are determined by an evaluation based on the forward-backward error, and these points are used to estimate the displacement of the bounding box.

2.4. **Boosting [8].** In this method, the tracking step is based on template tracking using a classifier. The classifier is evaluated at many possible positions in the search region in the next image frame. Given a confidence map, which consists of confidence values corresponding to the possible positions in the classifier evaluation, the most probable position of the target object is considered to be the position with maximum confidence. An online AdaBoost algorithm [19] updates the features of the classifier while tracking the object to cope with changes in the object's appearance. The online trained classifier uses the surrounding background as negative examples, and updating the classifier using the positive and negative samples allows the selection of the most discriminative features between the object and the background.

2.5. **Our approach.** As discussed above, the development of effective object tracking algorithms and their improvement in terms of processing speed is an important problem in computer vision. In practice, however, no single algorithm can be adequately applied to every possible scene; each algorithm has individual strengths and weaknesses. Given that various algorithms are available in OpenCV, an effective framework combining the advantages of individual algorithms can provide more reliable and stable tracking results and contribute to flexible adaptation to various scenes. Although this approach involves some limitations in terms of processing speed, its reliable results are useful for purposes that require high accuracy, such as log analysis. Furthermore, because the framework depicts the performance of each algorithm used, the performance evaluation results can be useful for planning object tracking systems. As described in the next section, the proposed framework is designed to take advantage of the strengths of each algorithm.

3. **Framework for Robust Object Tracking Using Multiple Algorithms.** This section proposes a framework for robust object tracking using multiple algorithms [20]. The basic concept is described at the beginning of this section, and a flowchart for the proposed framework is shown. In the subsequent subsections, the details of some processing stages in the flowchart are described. As described below, the framework effectively combines the results of several tracking algorithms using weighting coefficients, and maintains underperforming algorithms in expectation of their future utility.

3.1. **Centroid by a weighted average.** Figure 1 schematically shows the basic concept of combining some object tracking results to form an integrated tracking result. The figure shows two calculation methods to obtain the integrated result. In the figure for each calculation method, rectangles as bounding boxes show object tracking results (left) for three tracking algorithms 1, 2, and 3 in this example, and a rectangle in a thick black line

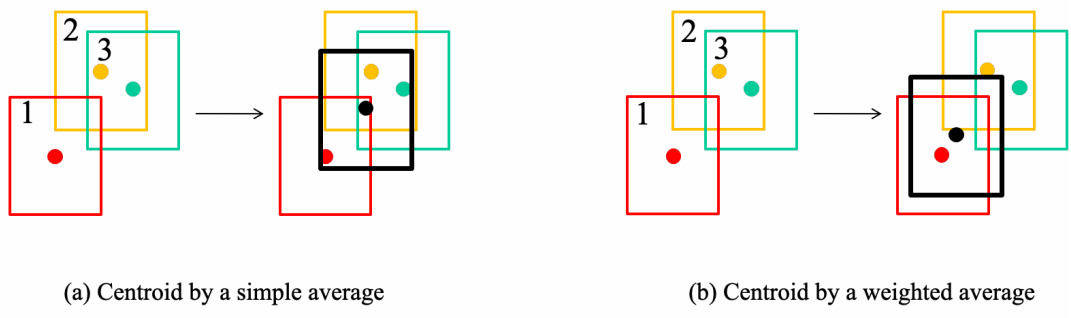

FIGURE 1. (color online) Calculation of the centroid of the rectangle as the integrated result (a) by a simple average and (b) by a weighted average; a greater weight is applied to the center point determined by algorithm 1, shown in red.

shows the integrated result (right). The center point of each rectangle is indicated by a solid circle. In Figure 1(a), the center point of the integrated result is obtained as the centroid based on a simple average of the center points of the object tracking results. In contrast, in Figure 1(b), the center point is obtained as the centroid based on a weighted average. In the example shown in Figure 1(b), because the object tracking result of an algorithm indicated as 1 has a larger weighting coefficient value than those of the other results, the center point of the integrated result is closer to that of result 1 than is the case in Figure 1(a). The size of the rectangle of the integrated tracking result can also be obtained by considering the tracking results of the algorithms.

In general, object tracking algorithms have particular strengths and weaknesses depending on the characteristics of a video sequence; therefore, tracking performance varies among algorithms. If tracking performance can be used to determine the values of the weighting coefficients, a more reliable tracking result can be expected using the centroid based on a weighted average, as shown in Figure 1(b). A method for determining the values of the weighting coefficients is described later.

3.2. **Framework.** Figure 2 shows a flowchart for the proposed framework. The framework mainly consists of target object selection, object tracking using OpenCV algorithms, centroid calculation, and preparation for the next tracking step. The flowchart shows the processing stages, with numbers to be used for reference in the explanation to follow, and supplementary information is added on the right-hand side to clarify the position of the OpenCV algorithms and the main process of the framework.

The overview of the process is as follows.

- Stage 1): A target object is selected to be tracked.
- Stage 2): After loading a new frame, OpenCV object tracking algorithms run in parallel and their tracking results are obtained.
- Stage 3): The tracking results are integrated into the result of the framework using the centroid calculation based on a weighted average, as shown in Figure 1(b).
- Stage 4): The value of the weighting coefficient for each algorithm is determined for the next tracking step.
- Stage 5): The rectangle positions determined by algorithms with poor tracking performance are adjusted so that these algorithms also participate in object tracking in the next tracking step.
- Stage 6): Object tracking continues by repeating Stage 2) through Stage 5) while more frames exist.

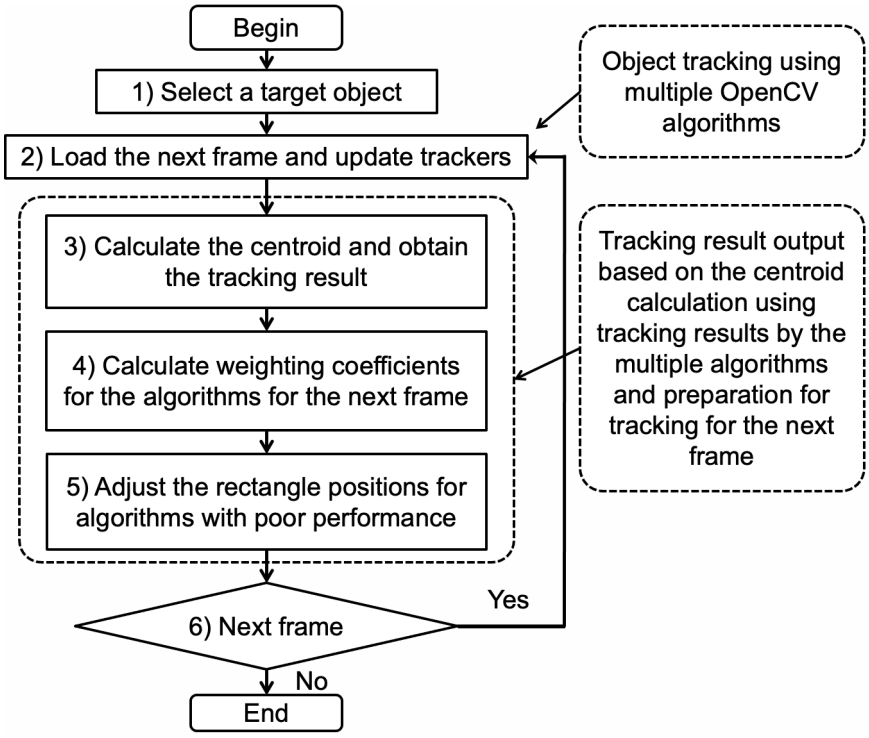

FIGURE 2. Flowchart for the proposed framework

As mentioned previously, four object tracking algorithms are used at Stage 2), namely KCF (Kernelized Correlation Filter), MIL (Multiple Instance Learning), Median Flow, and Boosting. At Stage 3), the size of the rectangle of the integrated tracking result is also updated by considering the tracking results of the algorithms. The details of Stages 4) and 5) are described in the subsequent subsections.

3.3. **Weighting coefficient.** The value of the weighting coefficient for each algorithm is determined at Stage 4), by considering the extent to which the tracking result from each algorithm covers the target object. As shown in Figure 1(b), the overlapped portion between the integrated rectangle (black) and the result of an algorithm is affected by the relative location of these rectangles. An algorithm with a larger overlapped portion can be considered to produce a more reliable tracking result relative to other algorithms, and therefore it can be expected to produce good performance for the next image frame. The candidate for the target object is obtained using the inter-frame difference method in the integrated rectangle. An area of 1.5 times larger than the integrated rectangle is used as the evaluation area for the inter-frame difference. Specifically, the value of the weighting coefficient is determined as the ratio of the number of pixels of the target object within the integrated rectangular area to those within the rectangular area corresponding to the algorithm. The weighting coefficients obtained in this way are used in Stage 3) for the next image frame.

3.4. **Rectangle-position adjustment.** Even if an algorithm has poor tracking performance for the current video sequence, it may have the potential to track the object successfully for future video sequences owing to changes in the image properties of the moving object.

Figure 3 schematically shows (a) the overview of the adjustment and (b) the criterion for applying the adjustment to the discrepancy. In Stage 5), tracking results from underperforming algorithms are adjusted, by repositioning the rectangles produced by those

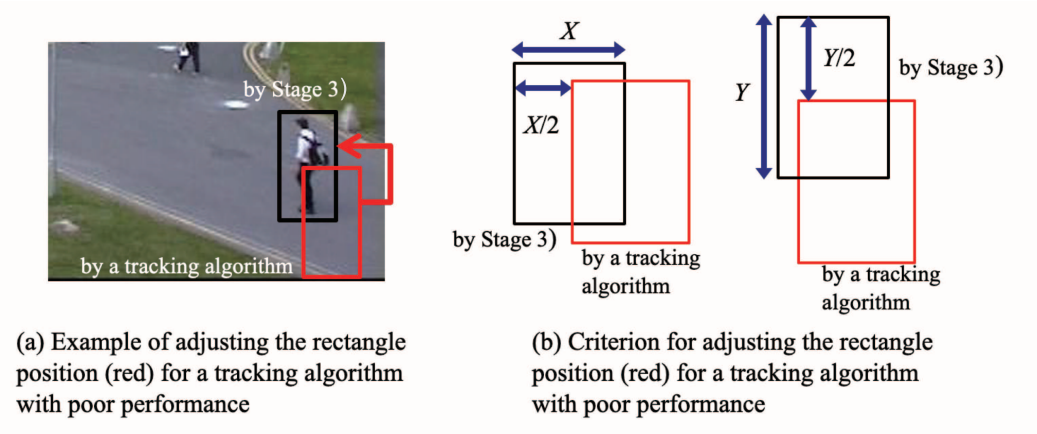

FIGURE 3. (color online) Schematic explanation for adjusting the position of a rectangle produced as the tracking result from an algorithm with poor performance. In part (a), the red rectangle indicates the original position, and the black rectangle is the integrated result from Stage 3) of the framework.

algorithms. This is done based on the position of the integrated result determined in Stage 3). As shown in Figure 3(b), a discrepancy beyond half the size of the rectangle for the integrated result is used as the criterion for applying the adjustment.

It is expected that the proposed framework, comprising the processing stages shown in Figure 2, will contribute to the further development of methods to effectively deal with the tracking results of several algorithms provided in OpenCV to achieve robust object tracking. Experimental results demonstrating the performance of the framework are presented in the next section, and the contributions of this work are described in further detail in the Discussion.

4. Experiments and Results.

To demonstrate the effectiveness of the framework proposed in the previous section, experiments were conducted for several video sequences.

In the experiments, the framework was developed on a laptop computer (CPU: Intel Core i3-4000M @ 2.40 GHz, memory: 4 GB, OS: Windows 7 Professional 32bit) using Visual Studio 2015 and OpenCV 3.2. To use tracking modules, OpenCV 3.2 was installed with opencv_contrib modules.

4.1. Result of a walking video sequence.

A video sequence "Walking" in Visual Tracker Benchmark v1.0 [21] was used for the test data. This sequence consists of 412 frames with an image size of 768 × 576 pixels.

This video sequence has the following characteristics.

- The camera is fixed and the background of the image does not move.
- The tracking target is a walking person, so the movement is slow.
- Since the target walks away from the camera, its size changes.

Some sample images are shown below.

In the following section, object tracking results are shown for the following four cases to evaluate the effects of the processing steps.

- Case 1: OpenCV object tracking algorithms are used independently.
- Case 2: The centroid by a simple average shown in Figure 1(a) is used.
- Case 3: The centroid by a weighted average shown in Figure 1(b) is used.
- Case 4: The centroid by a weighted average shown in Figure 1(b) is used and in addition, tracking results of algorithms with poor performance are adjusted.

In the following figures, 20 frames with an interval of 20 frames from the second frame are displayed as examples for each tracking result. The corresponding frame number is attached to the top right corner of each image frame.

Figure 4 shows the result for Case 1. The 20 frames are arranged from the top left to the bottom right. The target object walks from the bottom right to the middle left of the image. In each frame, the tracking results of KCF (Kernelized Correlation Filter), MIL (Multiple Instance Learning), Median Flow, and Boosting are shown as rectangles in green, magenta, red, and light blue, respectively. Although each of the four tracking algorithms in OpenCV can track the target object in the images in the first row of the array of images, Median Flow (in red) fails to track the object (frame No. 82) and its rectangle remains in the same position in the subsequent image frames. In the bottom row, KCF (in green) fails to track the object (No. 342), whereas the other algorithms MIL (in magenta) and Boosting (in light blue) track the object successfully. The success rates of tracking defined by the ratio of successful frames to total frames for KCF, MIL, Median Flow, and Boosting are 83%, 100%, 20%, and 100%, respectively.

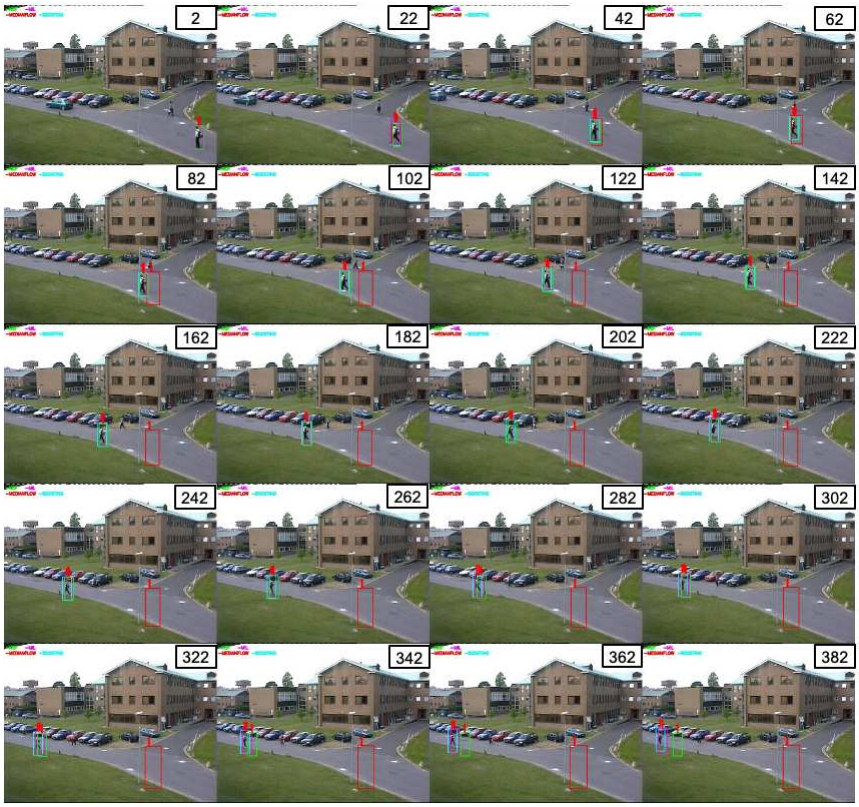

FIGURE 4. (color online) Result for Case 1 in which the OpenCV object tracking algorithms are used independently for "Walking". In each frame, the tracking results of KCF, MIL, Median Flow, and Boosting are shown as rectangles in green, magenta, red, and light blue, respectively. The top right corner of each frame indicates the corresponding frame number.

Figure 5 shows the result for Case 2. In this result, the black rectangle shows the centroid by a simple average of the tracking results of the four algorithms. Although the black rectangle can track the object in the first row, it fails to track the object (No. 122) for the second and successive rows. Because the centroid by a simple average of the tracking results is greatly affected by the failed tracking algorithms, stable and reliable tracking cannot be achieved.

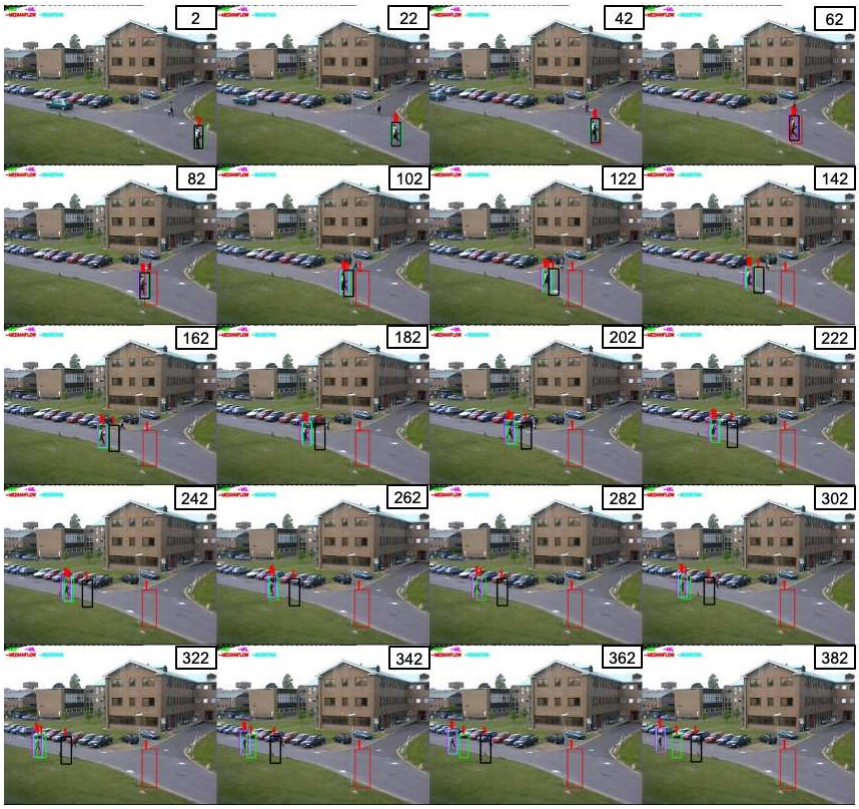

FIGURE 5. (color online) Result for Case 2 in which the centroid by a simple average is used, as shown in Figure 1(a)

Figure 6 shows the result for Case 3. In this result, the black rectangle shows the centroid by a weighted average of the tracking results of the four algorithms. As shown in the figure, the black rectangle can track the object successfully. Although Median Flow (in red) and KCF (in green) fail to track the object, their small weighting coefficient values and the large weighting coefficient values for the other algorithms provide a good integrated estimate of the position of the target object. This result suggests the usefulness of the weighting coefficients calculated based on the overlapped portions as shown in Section 3.3; the failed tracking algorithms do not affect the final tracking result.

Figure 7 shows the result for Case 4. In this case, the tracking results of algorithms with poor performance are adjusted. The positions estimated by the algorithms with poor performance were changed to the position of the centroid by a weighted average. As shown in Figure 7, the rectangle in black as the centroid and rectangles for the four algorithms overlap with the target object. In general, tracking algorithms have strengths and weaknesses depending on the scenes. As mentioned above for Case 1, the tracking success rates varied among the algorithms. Although there were differences in the tracking performance among the algorithms, the framework effectively combined all of their tracking results and succeeded in tracking all frames of the video sequence. Considering this situation, a comparison of the tracking result was performed between the four independently used OpenCV algorithms as shown in Case 1 in Figure 4 and the proposed framework as shown in Case 4 in Figure 7. The Euclidean pixel distances on the images between each result for Case 1 and the integrated result for Case 4 were evaluated. The distance was defined between the center point of the rectangle for each result and that of the integrated result. The averaged Euclidean pixel distances over all frames for KCF,

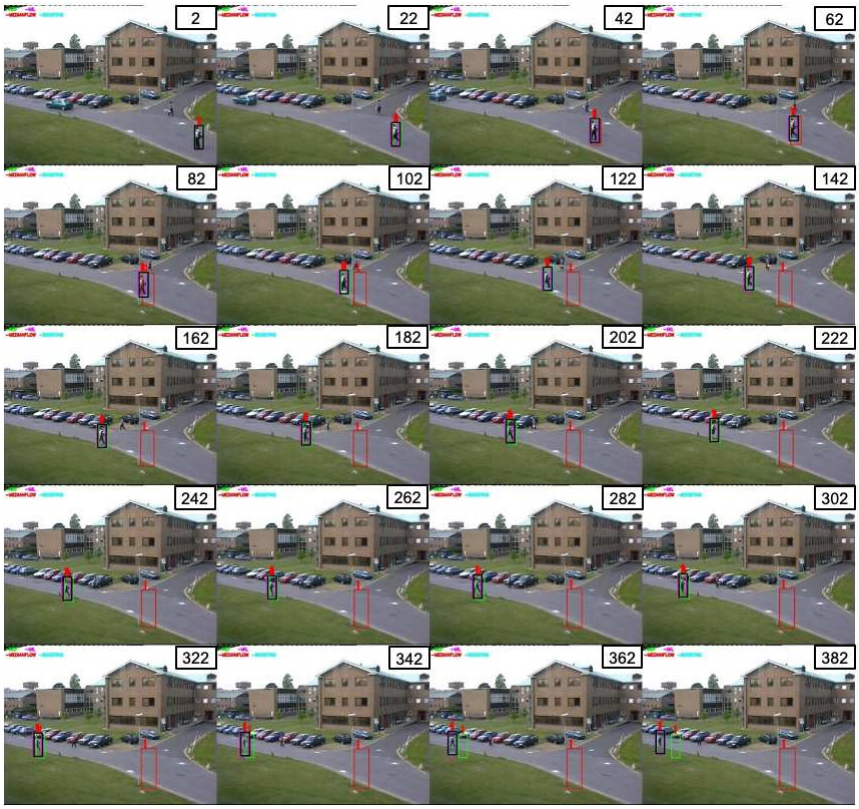

FIGURE 6. (color online) Result for Case 3 in which the centroid by a weighted average is used, as shown in Figure 1(b)

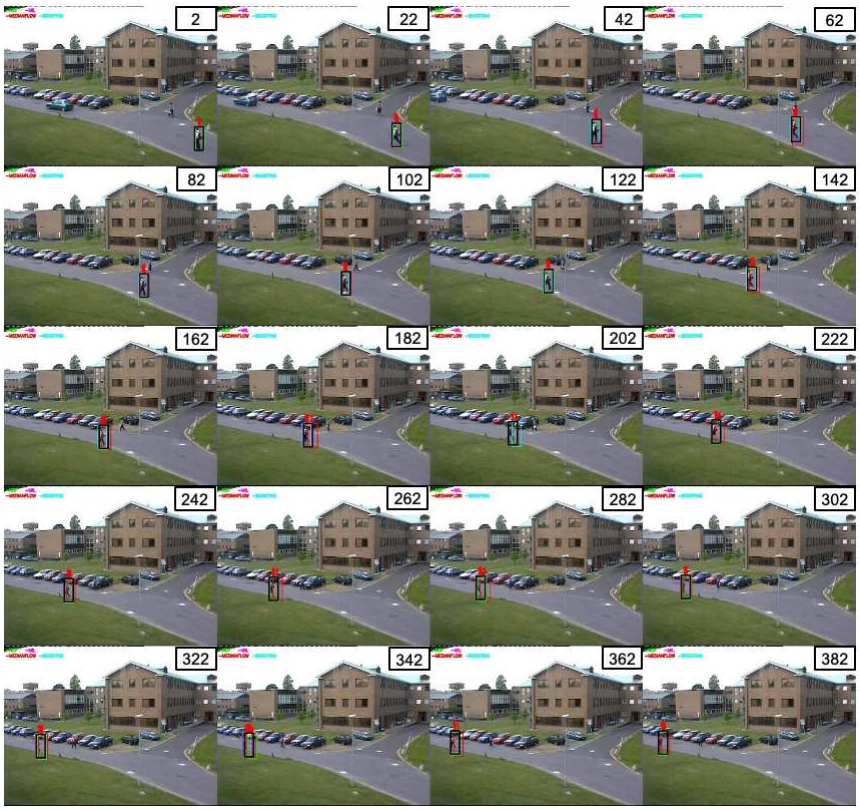

FIGURE 7. (color online) Result for Case 4 in which the tracking results of algorithms with poor performance are adjusted

MIL, Median Flow, and Boosting were 12.9, 3.9, 243.3, and 6.7 pixels, respectively. Large distance values reflect tracking failures of KCF and Median Flow in Figure 4.

Because all algorithms are available in Case 4, this method has the potential to apply to a wide range of scenes.

4.2. **Result of a basketball video sequence.** A video sequence "Basketball" in Visual Tracker Benchmark v1.0 [21] was used for the test data. This sequence consists of 725 frames with an image size of 576 × 432 pixels.

This video sequence has the following characteristics.

- The camera is moving and the background of the image changes.
- Large changes in body shape occur due to player actions.
- Some players move quickly.
- The target is sometimes occluded by other players.

Some sample images are shown below.

Object tracking results are shown for the following two cases to evaluate the effects of the related processing steps.

- Case 1: OpenCV object tracking algorithms are used independently.
- Case 2: The centroid by a weighted average shown in Figure 1(b) is used and in addition, tracking results of algorithms with poor performance are adjusted.

In the following figures, 16 frames with an interval of 48 frames from the second frame are displayed for each tracking result.

Figure 8 shows the result for Case 1. The 16 frames are arranged from the top left to the bottom right. The target object is a player wearing a green uniform around the center of the image in the initial frame. In each frame, the tracking results of KCF, MIL, Median Flow, and Boosting are shown as rectangles in green, magenta, red, and light blue, respectively. Although each of the four tracking algorithms can track the target object in the images for two frames (Nos. 2 and 50) arranged in the first row, most of the algorithms fail to track the object in the subsequent frames.

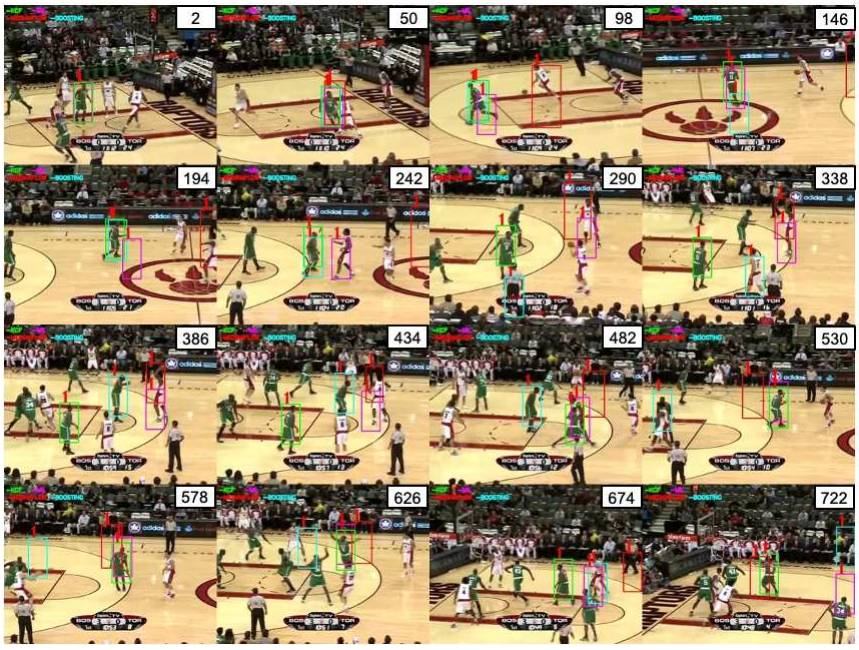

FIGURE 8. (color online) Result for Case 1 in which the OpenCV object tracking algorithms are used independently for "Basketball"

Figure 9 shows the result for Case 2. In this result, the black rectangle is based on the centroid by a weighted average of the tracking results of the four algorithms. The black rectangle can successfully track the object through all frames of the video sequence. Some tracking algorithms fail to track the object, and their rectangles are displayed in slightly different positions from the black rectangle for some frames. However, owing to the adjustment process for the tracking results of algorithms with poor performance, all algorithms are successful even in the subsequent frames.

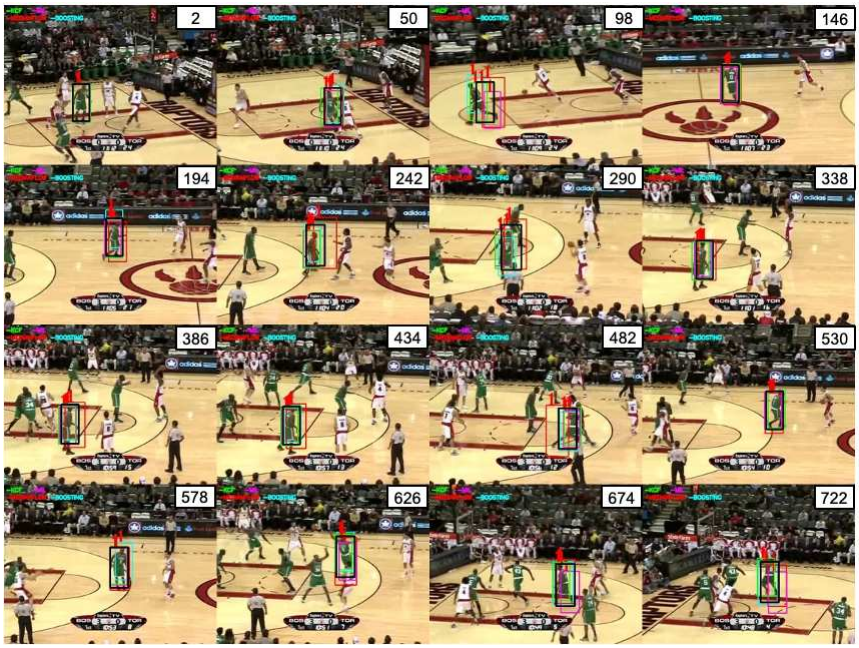

FIGURE 9. (color online) Result for Case 2 in which the centroid by a weighted average is used and in addition, tracking results of algorithms with poor performance are adjusted

As shown in Figure 9 for Case 2, the proposed framework successfully tracked the object through all the frames. Considering this situation, a comparison of the tracking result was performed between the four independently used OpenCV algorithms as shown in Case 1 in Figure 8 and the proposed framework as shown in Case 2 in Figure 9. Similar to the "Walking" example, the Euclidean pixel distances on the images between each result for Case 1 and the integrated result for Case 2 were evaluated. Figure 10 shows the distances of the four OpenCV algorithms as a function of the image frame number. Because the integrated result for Case 2 successfully tracked the target object, the large pixel distance in the graph represents a tracking failure. The four algorithms successfully tracked the target object until approximately the 70th frame. However, Median Flow (in red) failed to track the object, and subsequently MIL (in magenta) and Boosting (in light blue) also failed to track. In contrast, KCF (in green) maintained successful tracking performance. The averaged Euclidean pixel distances over all frames for KCF, MIL, Median Flow, and Boosting were 10.8, 94.7, 170.0, and 108.5 pixels, respectively.

Figure 11 shows a comparison of the 279th frame between Case 1 and Case 2. In the frame shown on the left, KCF (in green) tracked the target object successfully, whereas the other algorithms failed to track the object and instead tracked other players or a referee. An abrupt increase in the Euclidean pixel distance for the Boosting algorithm before the 300th frame in Figure 10 reflects this erroneous tracking. In contrast, in the

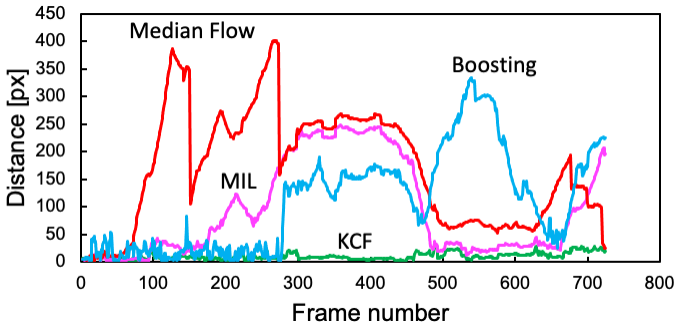

FIGURE 10. (color online) Euclidean pixel distances for the four OpenCV algorithms from the results of the proposed framework

(a) Case 1 (b) Case 2

FIGURE 11. (color online) Comparison of the 279th frame between Case 1 and Case 2

frame shown on the right, the resulting black rectangle based on the proposed framework successfully captured the target object.

This result suggests that the framework can use a combination of the advantages of each algorithm and avoid fatal tracking errors, and consequently, achieve successful and stable object tracking.

4.3. **Result of an ant video sequence.** An original video of an ant was used for test data. This sequence consists of 699 frames with an image size of 640 × 480 pixels, at a frame rate of 120 fps. A camera (CASIO EXILIM EX-100F, 12.1 million pixels) was used to record the video in MOV format, and the video was converted to individual frames in JPG format.

An example image is shown in Figure 12. An ant as a target object is located at the left in the image, surrounded by a black rectangle which shows the location of the selected

(a) Original image (b) Magnified image of (a)

FIGURE 12. (color online) Original image in the initial frame and a magnified image of an ant as a target. The white arrow indicates the location of the ant.

target in the initial frame. As can be seen, the target ant is very small, and most of the image is the bark of a rotten tree.

This video sequence has the following characteristics.

- The camera is fixed and the background of the image does not move.
- The target object is small in the image and difficult to see with the naked eye.
- The target object easily vanishes into bark and crevices.

Object tracking results are shown for the following two cases to evaluate the effects of the processing steps.

- Case 1: OpenCV object tracking algorithms are used independently.
- Case 2: The centroid by a weighted average shown in Figure 1(b) is used and in addition, tracking results of algorithms with poor performance are adjusted.

In the following figures, 16 frames with an interval of 40 frames from the second frame are displayed for each tracking result.

Figure 13 shows the result for Case 1. The 16 frames are arranged from the top left to the bottom right. In each frame, tracking results for KCF, MIL, Median Flow, and Boosting are shown as rectangles in green, magenta, red, and light blue, respectively. Except for the initial frame, Median Flow (in red) fails to track the object, whereas the other algorithms track the object successfully.

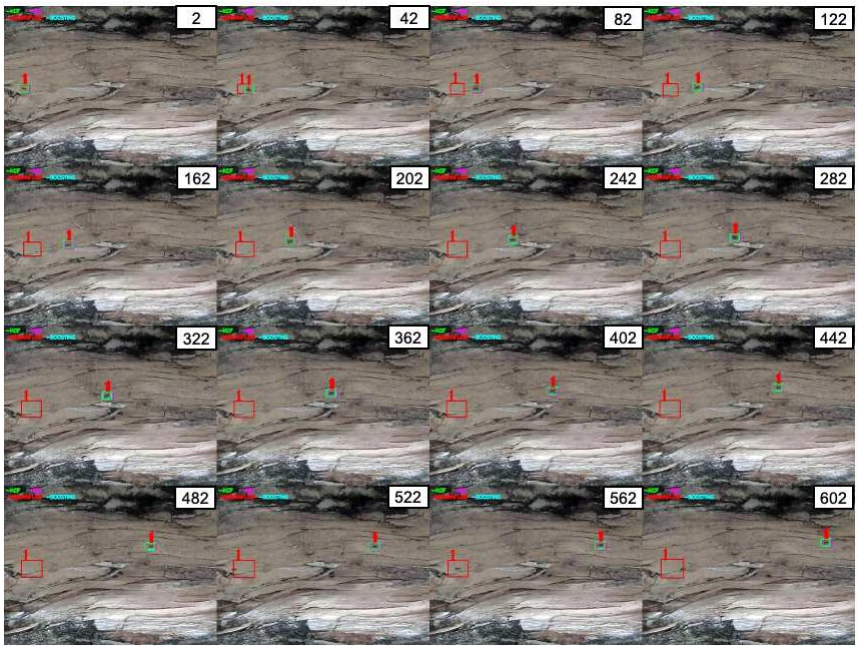

FIGURE 13. (color online) Result for Case 1 in which the OpenCV object tracking algorithms are used independently for "Ant"

Figure 14 shows the result for Case 2. In this result, the black rectangle shows the centroid by a weighted average of the tracking results of the four algorithms. As shown in the figure, the black rectangle can track the object successfully as it moves from the left to the right through all frames.

The averaged Euclidean pixel distances, which are defined in the same manner as the "Basketball" case, for KCF, MIL, Median Flow, and Boosting through all frames were 2.7, 5.7, 258.5, and 4.2 pixels, respectively. Except for the Median Flow algorithm, which failed to track the object, relatively small distance values suggest that successful object tracking was performed by the OpenCV algorithms and the proposed framework.

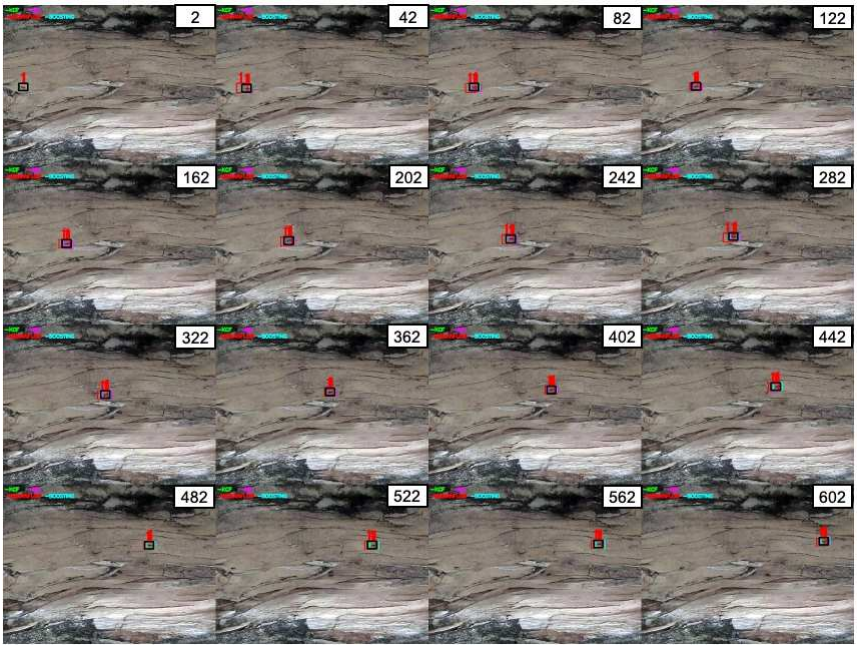

FIGURE 14. (color online) Result for Case 2 in which the centroid by a weighted average is used and in addition, tracking results of algorithms with poor performance are adjusted

5. **Discussion.** The previous section shows that the proposed framework tracked the object in each video example successfully, showing robust object tracking. Even in the case in which most of the independently used OpenCV algorithms failed to track the object, the framework provided reliable tracking results, as shown for the "Basketball" example. In this example, KCF showed outstanding performance among the four OpenCV algorithms as shown in Figure 10, whereas KCF failed to track the object in the frames (Nos. 362 and 382) in Figure 4 for the "Walking" example. These examples show that any given single algorithm may be expected to exhibit individual strengths and weaknesses, and suggest the usefulness of the proposed framework, which showed robust and stable tracking results.

The proposed framework provides us with the following benefits.

- Tracking results of multiple algorithms in OpenCV are effectively combined to create a tracker that captures a target object.
- Because all algorithms are ready for a new image frame even after some algorithms exhibited poor tracking performance for the previous frame, the framework maintains the strengths of each algorithm, and consequently, contributes to flexible adaptation to various scenes.
- Adding or removing algorithms is easily controlled using the flexible framework.
- The ability to use the OpenCV algorithms without modification is an advantage of the framework.
- The performance of individual OpenCV algorithms can be compared and the result can be used to select algorithms that suit the scene to be analyzed.

Although this framework requires greater computational resources than any single OpenCV tracking algorithm, the proposed framework is expected to provide flexible solutions to meet the demands of a variety of object tracking requirements.

6. **Conclusions.** This study proposes a framework for a robust object tracking method that combines multiple OpenCV algorithms. In the framework, tracking results from multiple algorithms are effectively combined, considering their individual performance, to create a rectangle tracker that captures a target object. The results of experiments using video sequences demonstrate that the proposed framework was able to track a target object successfully even in cases in which most of the OpenCV algorithms individually failed to do so.

Occlusion of target objects by surrounding objects is a challenging issue for object tracking. Used in combination with other predictive systems such as the Kalman filter, the proposed framework has the potential to overcome occlusion issues for a relatively long term, and further development of such a system is the next step for future work.

Acknowledgment. Part of this work was supported by Grants-in-Aid for Scientific Research JP17K06464 and JP21K04081 from the Japan Society for the Promotion of Science.

REFERENCES

[1] O. Elharrouss, N. Almaadeed and S. Al-Maadeed, A review of video surveillance systems, *Journal of Visual Communication and Image Representation*, vol.77, 103116, 2021.

[2] M. J. Black and A. D. Jepson, EigenTracking: Robust matching and tracking of articulated objects using a view-based representation, *International Journal of Computer Vision*, vol.26, no.1, pp.63-84, 1998.

[3] D. Comaniciu, V. Ramesh and P. Meer, Real-time tracking of non-rigid objects using mean shift, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.142-149, 2000.

[4] S. Avidan, Support vector tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.26, no.8, pp.1064-1072, 2004.

[5] J. F. Henriques, R. Caserio, P. Martins and J. Batista, High-speed tracking with kernelized correlation filters, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.37, no.3, pp.583-596, 2015.

[6] B. Babenko, M.-H. Yang and S. Belongie, Visual tracking with online multiple instance learning, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.983-990, 2009.

[7] Z. Kalal, K. Mikolajczyk and J. Matas, Forward-backward error: Automatic detection of tracking failures, *Proc. of the 20th International Conference on Pattern Recognition (ICPR)*, pp.2756-2759, 2010.

[8] H. Grabner, M. Grabner and H. Bischof, Real-time tracking via on-line boosting, *Proc. of the British Machine Vision Conference (BMVC)*, pp.6.1-6.10, 2006.

[9] Y. Hashimoto, H. Hama and T. T. Zin, Robust tracking of cattle using super pixels and local graph cut for monitoring systems, *International Journal of Innovative Computing, Information and Control*, vol.16, no.4, pp.1469-1475, 2020.

[10] H. Pradana and K. Horio, Automatic controlling fish feeding machine using feature extraction of nutriment and ripple behavior, *International Journal of Innovative Computing, Information and Control*, vol.17, no.5, pp.1483-1500, 2021.

[11] W. Zhang, B. Kang, S. Zhang and M. Gao, Visual object tracking with saliency refiner and adaptive updating, *International Journal of Innovative Computing, Information and Control*, vol.14, no.5, pp.1855-1875, 2018.

[12] *OpenCV*, https://opencv.org/about/, Accessed on Oct. 21, 2021.

[13] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.886-893, 2005.

[14] D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, Visual object tracking using adaptive correlation filters, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2544-2550, 2010.

[15] J. F. Henriques, R. Caseiro, P. Martins and J. Batista, Exploiting the circulant structure of tracking-by-detection with kernels, *Proc. of European Conference on Computer Vision (ECCV)*, pp.702-715, 2012.

[16] T. G. Dietterich, R. H. Lathrop and L. T. Perez, Solving the multiple instance problem with axis-parallel rectangles, *Artificial Intelligence*, vol.89, nos.1-2, pp.31-37, 1997.

[17] B. Lucas and T. Kanade, An iterative image registration technique with an application to stereo vision, *Proc. of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, pp.674-679, 1981.
[18] J. Shi and C. Tomasi, Good features to track, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.593-600, 1994.
[19] H. Grabner and H. Bischof, On-line Boosting and vision, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.260-267, 2006.
[20] K. Tanaka and K. Ogata, A study of a robust object tracking system with multiple algorithms using OpenCV, *Record of 2018 Joint Conference of Electrical, Electronics and Information Engineers in Kyushu*, p.273, 2018 (in Japanese).
[21] *Visual Tracker Benchmark v1.0*, http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html, Accessed on Oct. 21, 2021.

Author Biography

Kohichi Ogata is an Associate Professor with the Department of Computer Science and Electrical Engineering, Faculty of Advanced Science and Technology, Kumamoto University. He received the B. Eng., M. Eng. and Ph.D. degrees in Engineering from Kumamoto University, Kumamoto, Japan, in 1989, 1991, and 1994, respectively. His research interests include signal processing, speech processing, and image processing, such as the measurement of speech production process and development of its applications, development of eye-gaze interface systems, and augmented reality application systems.

Koki Tanaka received the B. Eng. and M. Eng. degrees in Engineering from Kumamoto University, Kumamoto, Japan, in 2017 and 2019, respectively. He is now with Sony Semiconductor Manufacturing Corporation. His research interests include image processing and signal processing.

Rinka Ikeda is a student in the Department of Computer Science and Electrical Engineering at Kumamoto University. Her research interests include image processing.

Fitri Utaminingrum is an Associate Professor in the Faculty of Computer Science, Brawijaya University, Indonesia. Currently, her focus research is about computer vision area, especially on the development of image algorithms. She is also as a full-time lecturer at Brawijaya University, Indonesia. She has been published her work in several reputable journals and conferences indexed by Scopus. She received a Bachelor's degree in Electrical Engineering field, and a master's degree in the same major from Brawijaya University, Malang, Indonesia. In addition, she obtained a Doctor of Engineering in the field of Computer Science and Electrical Engineering from Kumamoto University, Japan especially focused on Computer Vision and Image Media Processing.