Autonomous Tracking by an Adaptable Scaled *KCF* Algorithm

Din-Chang Tseng, Chien-Hung Chen, and Yi-Ming Chen

***Abstract*—A multicopter is equipped by a passive tracking device to follow a specified target. However, if want to track a non-controlled target, the passive tracking device is failed. We propose a vision-based tracking system for multicopters, used computer vision method to track any target without additional tracking devices. In this study, propose scale candidate graphs and scale tables to improve *KCF*. There are also stable results when the scale changes. The proposed an adaptable scaled *KCF* algorithm, when the *KCF* tracking failed, a feature-based matching detector is used to re-detect the target. Several experiments on various scene based on the proposed approach were conducted and evaluated. Stable tracking results were obtain to show the feasibility of the proposed system.**

***Index Terms*—Computer vision, image processing, object tracking, multicopters.**

I. Introduction

The applications of multicopters for aerial photography is most popular [1]. In general, the flying path of a multicopters is controlled by a remote controller, and manually controlling the multicopters to follow the moving target. Other equipped by a passive tracking device to follow a specified target. In order to conveniently track targets, a multicopters is equipped by a passive tracking device to follow a specified target. However, if we want to track a non-controlled target, the passive tracking device is failed. We propose a vision-based tracking system for multicopters, used computer vision method to track any target without additional tracking devices. The proposed system divided into three parts: generate reaction graphs, feature extraction and target matching. To do tracking, we must first be able to capture a target object in the image sequence.

Generate reaction graphs method, Zhang *et al.* [2] using radius sliding window with a radius at the position tracked by the previous frame. Kalal *et al.* [3] proposed a single-target long-term tracking algorithm for Tracking-Learning-Detection (*TLD*), after the target is selected in the first frame, a sliding window is used at the beginning of each frame. In addition to generating candidate regions by exhaustive methods, can use the characteristics of the target between frames and frames without moving too much, using Lucas Kanade optical-flow [4], Kalman filter [5] and Particle Filter algorithm to predict the result of the next frame as a candidate area.

Manuscript received November 11, 2019; revised August 15, 2020.

The authors are with The Institute of Computer Science and Information Engineering, National Central University, Jhongli, 32001, Taiwan (e-mail: chienhung66@gmail.com).

Feature tracking is to capture various information of a target in an image as feature information, Comaniciu and Meer [6] proposed a tracking method using mean shift to calculate the color distribution in the target region as the target feature model, and then use this color feature model to match each possible target region. Background subtraction [7], [8] usually requires background model training, using the trained background model as a feature to detect foreground images of next inputs. Kim *et al.* [7] proposed CB algorithm adopts a quantization/clustering technique, used color distortion and brightness distortion classified as background, and established multimode background model. This method can to encode moving backgrounds or multiple changing, and the capability of coping with local and global illumination changes backgrounds. Image features are important for the comparison of candidate images and target images. General image features are Features from accelerated segment test (*FAST*) [9], Harris Corner [10], Binary Robust Independent Elementary Features (*BRIEF*) [11], Scale invariant feature transform (*SIFT*) [12], Speeded-Up Robust Features (*SURF*) [13] and color histogram, the quality of the feature extraction affects the entire tracking result. Cheng *et al.* [14] proposed a tracking method combining Particle Swarm optimization (*PSO*) [15] and *SIFT*, using *PSO* algorithm to find candidate regions, and then integrating *SIFT* features into *PSO* results to obtain more accurate tracking results. Miao *et al.* [16] proposed an adaptive classifier tracking method to match the feature points between successive frames, using *SURF* as a feature point, with an adaptive online boosting [17] classifier, and a sample weighting mechanism to establish robust feature descriptions and reliable feature point matching for tracking. Leichter *et al.* [18] proposed an extension of the mean shift tracking, used a color distribution map obtained from multiple different frames to enhance the mean shift tracking. Make a convex hull on the color map and use it in the target model, this maintains the original convergence and speed of the mean shift tracking and can be successfully tracked when the target color changes dramatically.

Zhang *et al.* [2] proposed a real-time tracking algorithm based on compressed sensing. In order to achieve scale invariance, each sample convolved with a multi-scale rectangular filter to obtain high-dimensional multi-scale features, but high-dimensional multi-scale features can cause excessive computation, while sparse perceptual theory can compress images and preserve original features. Therefore, using sparse random matrices to reduce multi-scale image features, this can speed up feature capture and achieve real-time tracking. Kalal *et al.* [19] proposed a

doi: 10.18178/ijmlc.2021.11.1.1013

forward-backward error method, use optical flow to find the possible position of the target in each frame. The features used by Kalal *et al.* [3] for *TLD* are similar to the local binary pattern (*LBP*) [20] and use the trained classifier to find out the probability that this feature may be the target. Bolme *et al.* [21] proposed the minimum output sum of squared error (*MOSSE*), using the correlation filter (*CF*) template as the target feature to measure the similarity for each input image and template image, the highest similarity is the target.

After the feature captured, the target founded from a plurality of candidate regions. In this step, the classifier often used to calculate the probability of the target in each candidate region, as Support Vector Machine *(SVM)* [22] and AdaBoost [23]. Lu *et al.* [24] used *PSO* to replace the traditional sliding window search for tracking, and put the *RGB* values of the corresponding window of the particle as features to AdaBoost to calculate the probability that the particle belongs to the target. Integrating all particles whose probability exceeds the threshold is the target. Zhu *et al.* [25] proposed a feature matching method. Use the extracted Harris corner to find the affine transformation of the image between frames, then use *SVM* to check and remove the unmatched feature points. Babenko *et al.* [26] combine online boosting with multiple instance learning (*MIL*) to train target models. In this paper, we use the Kernel Correlation Filter (*KCF*) proposed by Henriques *et al.* [27] for target tracking, but *KCF* cannot adapt to target scale changes, so we propose scale candidate graphs and scale tables to improve *KCF*. There are also stable results when the scale changes. The proposed an adaptable scaled *KCF* algorithm, when the *KCF* tracking is failed, a feature-based matching detector is then used to re-detect the target.

This paper is structured as follows: The details of the proposed techniques for use in the system are presented in Sections in Section II. Experimental results are included in Section III, followed by conclusions in Section IV.

II. Proposed Techniques

The tracking module by an adaptable scaled *KCF* algorithm are described in detail below.

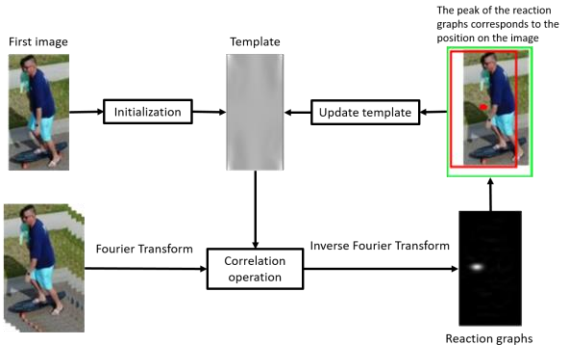


Fig. 1. Correlation filter applied to target tracking.

A. Correlation Filter

The minimum Output Sum of Squared Error filter (*MOSSE*) which applies correlation filtering to target tracking, as shown in Fig. 1, and evaluate the correlation of the two signals. If the two signals are similar, the correlation is higher. It is necessary to generate a filter template for tracking, let the target can generated the maximum response on the template, and the position of the maximum response value is the target position.

When tracking initialization, need to generate a filter template to maximize the target's response on the template, and can be written as

$$g = f \otimes h \tag{1}$$

where g is reaction graphs, h is filter template and f is output image.

Assume the reaction graphs is designed to be Gaussian shape, as shown in Fig. 2, after knowing the reaction graph and the input image, the template is required. The correlation filter can be calculated at high speed because the Fast Fourier Transform (*FFT*) is used in the operation.

$$Fg = F(f \otimes h) = F(f) \cdot F(h)^* \tag{2}$$

Let $Fg = G$, $Ff = F$, $F(h)^* = H^*$ and can be described as

$$G = F \cdot H^* \tag{3}$$

Refer to multiple images for application to make the template more robust, so described as

$$\min_{H^*} \sum_i |F_i \cdot H^* - G_i|^2 \tag{4}$$

Solving equation (4) to get the template H,

$$H = \frac{\sum_{i=1}^{m} F_i \cdot G_i^*}{\sum_{i=1}^{m} F_i \cdot F_i^*} \tag{5}$$

After getting the template, then can start tracking. Only need to correlate the input image after *FFT* with the template to get the reaction graph. Then use the Inverse Fast Fourier Transform (*IFFT*) to find the peak position. The position of the peak of the reaction graph is the position of the target.

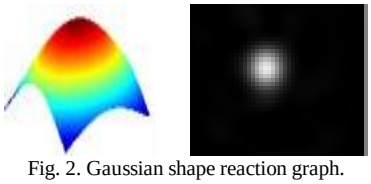


Fig. 2. Gaussian shape reaction graph.

In order to make the tracking results better, the template must updated as the target changes. If the template remains the same forever, the tracking will fail when the target changes to a certain extent. The update method is

$$h_t = (1 - \sigma)\, h_{t-1} + \sigma\, h_t \tag{6}$$

where h_t is determined filter template at t time, h_{t-1} is determined filter template at t+1 time, σ is empirical constant.

B. Generate Candidate Graphs

The original *KCF* does not have the ability to adapt to the

target scale change. Target cannot be correctly selected when the scale change. As a result, it will be selected the background or unable to select the entire target, as shown in Fig. 3. Therefore, we propose scale candidate graphs and scale tables to improve this problem.

Fig. 3. Target cannot be correctly selected when the scale change.

Because the template size of the *KCF* algorithm cannot be change, so we change the size of the input image to approximate the template. The principle is that the camera shoots objects at different distances. The photosensitive element of the camera as our template cannot be resize, so the object needs to adjust the focal length of the lens when shooting different distances, let the object is just inserted into the photosensitive element through the lens. Lens zoom is our scale table, as shown in Fig. 4.

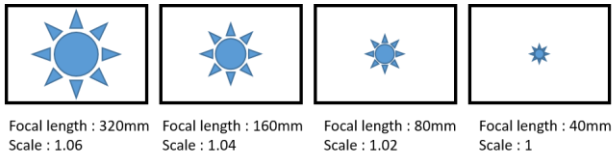


Fig. 4. Camera principle.

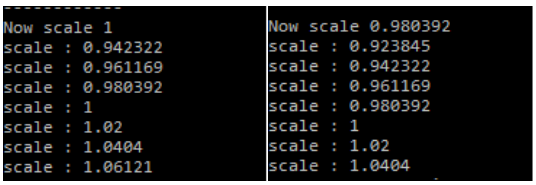


Fig. 5. The scale table.

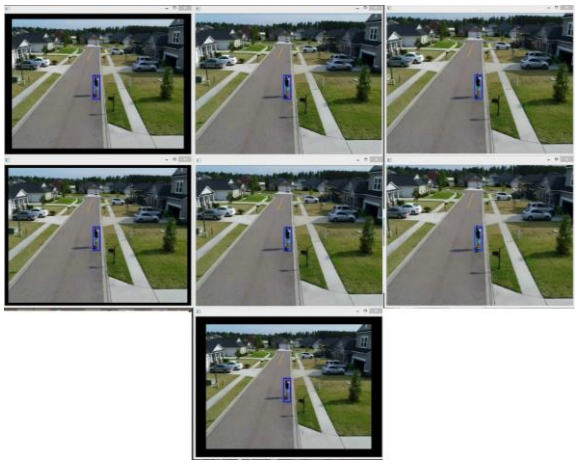


Fig. 6. Seven scales candidate graphs.

Using the scale table, as shown in Fig. 5, the candidate graphs with seven different scales of the input image, as shown in Fig. 6. Let the target size in the input image be close to the template size, and then perform *KCF* tracking on the seven candidate images. In different scales of input images, the target size is closer to the template have a higher similarity, as shown in Fig. 7. Therefore, the highest similarity of the tracking results is the target.

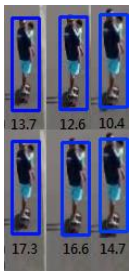


Fig. 7. The similarity obtained by *KCF* for different target sizes.

The candidate graph and the template are correlate to obtain a reaction graph. The template correlation of the reaction graph is calculate by equation (7). The template correlation is the similarity between the candidate graph and the template, and the template correlation are calculated as

$$\text{Tcorrelation} = R_{peak} \times \alpha + \frac{R_{peak} - R_{avg}}{R_{sd}} \times (1 - \alpha) \tag{7}$$

Obtained seven template correlations, with the highest value as the best template correlation, if the optimal template correlation is higher than the threshold, the tracking is successful, and the peak position of the reaction map corresponding to the correlation is the target position. If the optimal template correlation is lower than the threshold, the tracking fails. The criteria for tracking results as

$$\begin{cases} \text{Success}, & \text{Tcorrelation} > \text{Cthreshold} \\ \text{Fail}\ , & \text{Tcorrelation} < \text{Cthreshold} \end{cases} \tag{8}$$

In addition, the target set used to store the last 20 high-correlation target images, as shown in Fig. 8. When the peak of the response graph is higher than the threshold, the tracking result place into target set to replace the oldest target, and update the scale table. The target update criterion as

$$R_{peak} > \beta \tag{9}$$

where Tcorrelation is template correlation, α is empirical constant of template correlation, $Rpeak$ is peak of the reaction graphs, $Ravg$ is average value of the reaction graphs, Rsd is standard deviation of the reaction graphs, Cthreshold is template correlation threshold, β is target update threshold.

Fig. 8. The last 20 high-correlation target images.

C. Re-detection of Target Disappearance

Our feature point detection uses *FAST*, and the

characterization uses 64-dimensional *SURF*. Target re-detection divided into six step, (*i*) Find the *FAST* corner of each target and the entire image in the target set. (*ii*) Calculate the *SURF* description for each *FAST* corner point. (*iii*) Euclidean distance as a basis for *SURF* feature matching. (*iv*) Filter bad match points (based on matching point distance). (*v*) Obtain target scale and position. (*vi*) Reset *KCF* scale and position.

D. Feature Matching

Feature matching, first needs to detect the target with *FAST*, as shown in Fig. 9, and the feature points of the input image, as shown in Fig. 10. Then, the *SURF* description of each feature point is calculated, and generated array matching points, as shown in Fig. 11, based on the use of *Euclidean* distance as *SURF* feature matching. In order to make the matching result more stable, it is necessary to filter the matching points, as shown in Fig. 12.

Fig. 9. Feature points of the target image.

Fig. 10. Feature points of the input images.

(a) (b)

Fig. 11. Filter the matching points. (a) Matching points before filtering. (b) Matching points after filtering.

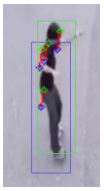


Fig. 12. Distance of matching points. (The blue point is the feature point on the target, the green point is the feature point of the input image, and the red line is the matching point distance).

E. Target Position and Scale Re-detection

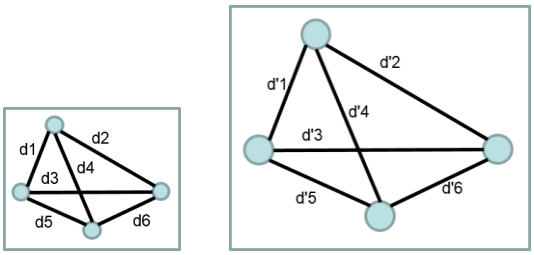
Calculate target displacement and scale using filtered matching points, and find the mode of the displacement vector between the matching points, which is the displacement vector of the target. The scale of *t* time estimated by the ratio of the sum of the distances between the feature points d_{sum} at *t*-1 time and the sum of the distances between the feature points d'_{sum} at time *t* time.

$$d'_{sum} = d'1 + d'2 + \ \ldots \ + d'6 \tag{10}$$

$$d_{sum} = d1 + d2 + \ \ldots \ + d6 \tag{11}$$

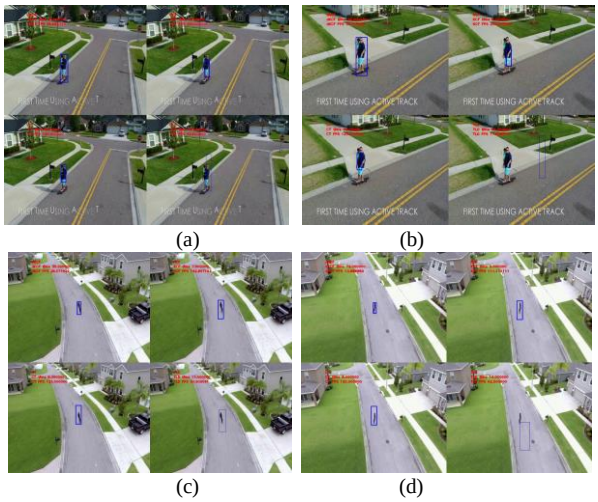
$$s_t = s_{t-1} \times \frac{d'_{sum}}{d_{sum}} \tag{12}$$

where d_{sum} is sum of the distances between the feature points at *t*-1 time, d'_{sum} is sum of the distances between the feature points at *t* time, s_t is target scale at *t* time, s_{t-1} is target scale at *t*-1 time. Last updated *KCF* scale and displacement, as shown in Fig. 13.



(a) (b)

Fig. 13. Scale estimate. (a) Feature points at *t*-1 time. (b) Feature points at *t* time.



(a) (b)

(c) (d)

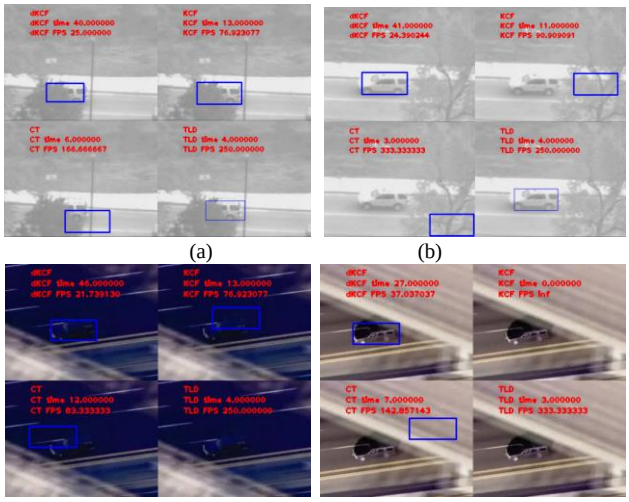
Fig. 14. Experimental results of two scaling scene. (a) and (b) is skateboard scene No.1. (c) and (d) is skateboard scene No.2. Each frame is the result of four algorithms. The upper left is the algorithm we proposed. The upper right is the *KCF* algorithm, and the lower left is the *CT* algorithm. The lower right is the *TLD* algorithm.

III. Experimental Results

In this section, we will show and compare experimental results of our proposed algorithm detectable *KCF* (*dKCF*), original *KCF*, Tracking-Learning-Detection (*TLD*) and Compressive Tracking (*CT*) methods, whether it can adapt to scaling and shadowing. When the target size changed, whether each methods can accurately select the target and when the target obscured, whether each methods can retrieve the target again. In the scaling scene, the results of our algorithm can be change and select target when the target size changed, as shown in Fig. 14, and in the shadowing scene, when the target disappears, our algorithm can re-track

to the target, it will not cause the tracking to fail, as shown in Fig. 15.

In Fig. 14, when the target size changes, our proposed algorithm can accurately selected the target, because we use the scale candidate map to generate different sizes of input images, so that the target size in the input image can be close to the template size, and the similarity will be the highest. Then take the input image of the best scale as the result. The target frame size of the algorithms *CT* and *KCF* will always remain the same, so the target cannot accurately selected. Instead, the frame box too large to cause the background to select, or the frame box too small to cause not to cover the target. If the target cannot accurately selected (too much background information or only a part of the target image information), the final tracking will fail.



(a) (b) (c) (d)

Fig. 15. Experimental results of two shadowing scene. (a) and (b) is car scene No.1. (c) and (d) is car scene No.2. Each frame is the result of four algorithms. The upper left is the algorithm we proposed. The upper right is the *KCF* algorithm, and the lower left is the *CT* algorithm. The lower right is the *TLD* algorithm.

In the target tracking, it may encounter the problem that the target obscured or disappeared. In order to solve this problem, we use the target set to record the target image. When the target obscured or disappeared, the feature matching used to find out whether there is an image similar to the target in the input image, and the most similar image is the target. In the other three algorithms (*KCF* algorithm, *CT* algorithm and *TLD* algorithm), only *TLD* algorithm has the ability to process the occlusion problem, the *KCF* algorithm and *CT* algorithm cannot retrieve the target after encountering the target occlusion.

The results of the experiment can divided into two cases, the target is shaded and scaled. There are total of four test films, each with 500 frames. The background of the film includes highways, general roads, trees, grasslands, shadow changes, etc. The success of the tracking based on manually selecting the target position in the film, then selecting the target using the algorithm, and finally comparing the results of selected by the manual and algorithm. In order to evaluate the ability of different algorithms, the target tracking success can be determined as positive, tracking failure determined as negative, and tracking success is based on

$$\begin{cases} \text{True}, \dfrac{|gt \cap bb|}{|gt \cup bb|} < 0.7 \ and \ \dfrac{|gt \cap bb|}{bb} < 0.5 \\ \text{False}, \dfrac{|gt \cap bb|}{|gt \cup bb|} \geq 0.7 \ and \ \dfrac{|gt \cap bb|}{bb} \geq 0.5 \end{cases} \tag{13}$$

where *gt* is manual selected target per frame, *bb* is the algorithm selected target per frame.

If the manual frame selection position overlaps with the algorithm position selected more than 70 percent and the overlap area accounts more than 50 percent of the frame selection area of the algorithm, and the target successfully tracked.

The performance of detection method can addressed by estimating the following parameters: (*i*) True Positive (*TP*) rate, (*ii*) False Negative (*FN*) rate, (*iii*) False Positive (*FP*) rate, and (*iv*) True Negative (*TN*) rate. Then can define the accuracy rate, tracking rate, and false positive rate. The experimental results of system assessment four video, each video length is 500 frames, the accuracy rate, tracking rate and false positive rate results of *TLD*, *CT*, *KCF*, and *dKCF* algorithm, as shown in Table I, Table II, and Table III. The average accuracy rate is 92.75 percent, the average tracking rate is 93.25 percent and the average false positive rate is 13.5 percent in our proposed algorithm.

TABLE I: THE ACCURACY RATE OF *TLD*, *CT*, *KCF*, AND *dKCF* ALGORITHM

Algorithm	Frames	Skateboard scene No.1	Skateboard scene No.2	Car scene No.1	Car scene No.2
TLD	500	74%	74%	37%	96%
CT	500	54%	68%	0%	0%
KCF	500	68%	78%	72%	45%
dKCF	500	78%	98%	96%	99%

TABLE II: THE TRACKING RATE OF *TLD*, *CT*, *KCF*, AND *dKCF* ALGORITHM

Algorithm	Frames	Skateboard scene No.1	Skateboard scene No.2	Car scene No.1	Car scene No.2
TLD	500	74%	74%	39%	97%
CT	500	54%	68%	24%	0%
KCF	500	68%	78%	100%	99%
dKCF	500	78%	98%	98%	99%

TABLE III: THE FALSE POSITIVE RATE OF *TLD*, *CT*, *KCF*, AND *dKCF* ALGORITHM

Algorithm	Frames	Car scene No.1	Car scene No.2
TLD	500	73%	0.7%
CT	500	100%	100%
KCF	500	100%	100%
dKCF	500	17%	10%

In addition to using the tracking status table to evaluate, we also compare the overlapping accuracy of the target selected, and the calculation equation for the overlap accuracy of each frame as

$$\begin{cases} 0 & , \dfrac{|gt \cap bb|}{|gt \cup bb|} = 0 \quad \text{or} \dfrac{|gt \cap bb|}{|gt \cup bb|} < 0.7 \\ \dfrac{|gt \cap bb|}{|gt \cup bb|} & , \dfrac{|gt \cap bb|}{|gt \cup bb|} \geq 0.7 \text{ and} \dfrac{|gt \cap bb|}{bb} \geq 0.7 \\ \dfrac{|gt \cap bb|}{|gt \cup bb|} - \left(1 - \dfrac{|gt \cap bb|}{bb}\right) & , \dfrac{|gt \cap bb|}{|gt \cup bb|} \geq 0.7 \text{ and} \dfrac{|gt \cap bb|}{bb} < 0.7 \end{cases} \tag{14}$$

where *gt* is manual selected target per frame, *bb* is the algorithm selected target per frame.

The overlap accuracy rate is to evaluate whether the algorithm can accurately selected target and calculate the area of each algorithm selected and manual selected. If the manual selected and algorithm selected have no overlap or the overlap area is less than 70 percent, the overlap accuracy rate is zero percent, conversely, there is overlap and area is higher than 70 percent then the overlap accuracy is the overlap area. Finally, the average overlap accuracy per frame is taken as the result and the average overlap accuracy is 87.25 percent in our proposed algorithm, as shown in Table IV.

TABLE IV: THE ALGORITHM EXECUTION SPEED AND OVERLAP ACCURACY OF *TLD*, *CT*, *KCF*, AND *dKCF* ALGORITHM

Algorithm	FPS	Skateboard scene No.1	Skateboard scene No.2	Car scene No.1	Car scene No.2
TLD	386	50%	54%	61%	56%
CT	195	51%	56%	33%	0%
KCF	145	60%	80%	41%	73%
dKCF	26	80%	88%	93%	88%

IV. CONCLUSION

In this paper, we have proposed *dKCF* algorithm allows multi-axis aircraft to have the ability to track non-specific targets through computer vision tracking. Because *KCF* is a template-based tracking method, there is no need to train the target in advance to achieve the goal of tracking non-specific targets. In tracking part, use the scale candidate graph to improve the *KCF* so that the *KCF* can adapt to the target change size and get the correct target image information. In re-detect part, find the *FAST* corner point on the input image target, calculate the *SURF* feature of each corner point, find the matching corner points as the matching points, and use the matching points to find the target position and scale to successfully retrieve the target. In the matching point filter part, use the mode of the displacement of the matching point as the matching criterion, such a standard is simpler and faster. In experimental, the proposed algorithm performs speed to 26 *fps*, tracking rate in the scaled cases can reach 88%, tracking rate in occluded cases can reach 98%, and overlap rate can reach 87%. With trade-off between speed and accuracy, we sacrifice the execution speed of the algorithm to exchange for more robust tracking. The future, system can be added color as an auxiliary feature, or create a more robust target model to complement the lack of template updates to increase the tracking effectiveness.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Din-Chang Tseng were in charge of overall direction and supervised the project; Chien-Hung Chen and Yi-Ming Chen carried out the experiments and concuted the research; Chien-Hung Chen wrote the paper; all authors discussed the results and contributed to the final manuscript and had approved the final version.

REFERENCES

[1] X. H. Dai, Q. Q., J. R. Ren, and K.-Y. Cai, "An analytical design-optimization method for electric propulsion systems of multicopter UAVs with desired hovering endurance," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 1, pp. 228-239, Feb. 2019.

[2] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. European Conf. on Computer Vision*, Firenze, Italy, Oct. 7-13, 2012, vol. 7574, no. 3, pp. 864-877.

[3] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409-1422, 2012.

[4] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int. Joint Conf. on Artificial Intelligence*, Vancouver, British Columbia, Canada, Aug. 24-28, 1981, pp. 674-679.

[5] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35-45, 1960.

[6] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603-619, 2002.

[7] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172-185, 2005.

[8] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Fort Collins, CO, Jun. 23-25, 1999, vol. 2, pp. 246-252.

[9] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. European Conf. on Computer Vision*, Graz, Austria, May 7-13, 2006, vol. 3951, pp. 430-443.

[10] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. Alvey Vision Conf.*, Oxford, UK, Aug. 31-Sep. 2, 1988, pp. 147-151.

[11] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: binary robust independent elementary features," in *Proc. European Conf. on Computer Vision*, Hersonissos, Greece, Sept. 5-11, 2010, vol. 6314, no. 4, pp. 778-792.

[12] D. G. Lowe, "Distinctive image features from scale invariant keypoints," *Int. Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[13] H. Bay, T. Tuytelaars, and L. VanGool, "SURF: Speeded up robust features," in *Proc. European Conf. on Computer Vision*, Graz, Austria, May 7-13, 2006,vol. 3951, pp. 404-417.

[14] X. Cheng, N. Li, S. Zhang, and Z. Wu, "Robust visual tracking with SIFT features and fragments based on particle swarm optimization," *Circuits, Systems, and Signal Processing*, vol. 33, no. 5, pp. 1507-1526, 2014.

[15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Conf. on Particle Swarm Optimization*, Perth, Australia, Nov. 27-Dec. 1, 1995, vol. 4. pp. 1942-1948.

[16] Q. Miao, G. Wang, C. Shi, X. Lin, and Z. Ruan, "A new framework for on-line object tracking based on SURF," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1564-1571, 2011.

[17] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, New York, NY, Jun. 17-22, 2006, vol. 1, pp. 260-267.

[18] I. Leichter, M. Lindenbaum, and E. Rivlin, "Mean shift tracking with multiple reference color histograms," *Computer Vision and Image Understanding*, vol. 114, no. 3, pp. 400-408, 2010.

[19] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: automatic detection of tracking failures," in *Proc. IEEE Conf. on Pattern Recognition*, Istanbul, Turkey, Aug. 23-26, 2010, pp. 2756-2759.

[20] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971-987, 2002.

[21] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, San Francisco, CA, Jun. 13-18, 2010, pp. 2544-2550.

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol.20, no. 3, pp. 273-297, 1995.

[23] Y. Freund and R. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," *Computer and System Sciences*, vol. 55, no. 1, pp. 119-139, 1995.

[24] H. Lu, W. Zhang, F. Yang, and X. Wang, "Robust tracking based on PSO and on-line AdaBoost," in *Proc. IEEE Conf. on Intelligent Information Hiding and Multimedia Signal Processing*, Kyoto, Japan, Sep. 12-14, 2009, pp. 690-693.

[25] W. Zhu, S. Wang, R.-S. Lin, and S. Levinson, "Tracking of object with SVM regression," in *Proc*. *IEEE Conf. on Computer Vision and Pattern Recognition*, Kauai, Hawaii, Dec. 8-14, 2001, vol. 2, pp. 240-245.
[26] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc*. *IEEE Conf. on Computer Vision and Pattern Recognition*, Miami, FL, Jun. 20-25, 2009, pp. 983-990.
[27] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters, " *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583-596, 2015.

Din-Chang Tseng received his Ph.D. degree in computer science and information engineering from National Chiao-Tung University, Hsinchu, Taiwan, in June 1988. He has been a professor in the Department of Computer Science and Information Engineering at National Central University, Jhongli, Taiwan since 1996. He is a member of the *IEEE*. His current research interests include computer vision, image processing, and virtual reality; especially in the topics of computer vision techniques for advanced driver assistance systems and human computer interaction.

Chien-Hung Chen received his M.S. degree in electrical engineering from National Chung Cheng Institute of Technology, Taoyuan, Taiwan, in 2003. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Information Engineering at National Central University, Jhongli, Taiwan. His research interests include computer vision, image processing, and real-time data processing.

Yi-Ming Chen received his B.S. degree in the Department of Computer Science and Information Engineering from Feng Chia University, Taichung, Taiwan, in 2015, and M.S. degree in the Institute of Computer Science and Information Engineering from National Central University, Jhongli, Taiwan, in 2017.