Ji et al. EURASIP Journal on Image and Video Processing (2021) 2021:21
https://doi.org/10.1186/s13640-021-00561-7

EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Real-time embedded object detection and tracking system in Zynq SoC

Qingbo Ji, Chong Dai, Changbo Hou* and Xun Li

*Correspondence:
Houchangbo@hrbeu.edu.cn
[1]College of Information and Communication Engineering, Harbin Engineering University, Harbin, China

Abstract

With the increasing application of computer vision technology in autonomous driving, robot, and other mobile devices, more and more attention has been paid to the implementation of target detection and tracking algorithms on embedded platforms. The real-time performance and robustness of algorithms are two hot research topics and challenges in this field. In order to solve the problems of poor real-time tracking performance of embedded systems using convolutional neural networks and low robustness of tracking algorithms for complex scenes, this paper proposes a fast and accurate real-time video detection and tracking algorithm suitable for embedded systems. The algorithm combines the object detection model of single-shot multibox detection in deep convolution networks and the kernel correlation filters tracking algorithm, what is more, it accelerates the single-shot multibox detection model using field-programmable gate arrays, which satisfies the real-time performance of the algorithm on the embedded platform. To solve the problem of model contamination after the kernel correlation filters algorithm fails to track in complex scenes, an improvement in the validity detection mechanism of tracking results is proposed that solves the problem of the traditional kernel correlation filters algorithm not being able to robustly track for a long time. In order to solve the problem that the missed rate of the single-shot multibox detection model is high under the conditions of motion blur or illumination variation, a strategy to reduce missed rate is proposed that effectively reduces the missed detection. The experimental results on the embedded platform show that the algorithm can achieve real-time tracking of the object in the video and can automatically reposition the object to continue tracking after the object tracking fails.

Keywords: Detection and tracking, Embedded, Real-time, Robustness, Strategy to reduce missed rate, Validity detection mechanism

1 Introduction

Object tracking has always been a focus of research in the field of computer vision. With the in-depth study of visual tracking algorithms by researchers, their scientific and theoretical basis has continued to improve, which has greatly promoted the development of surveillance systems, perceptual user interface, intelligent robotics, vehicle navigation, and intelligent transportation systems [1–3]. Two important factors to consider when

designing an embedded tracking system are the real-time performance of the algorithms and their robustness [4].

Due to the complexity of the scenes, object tracking algorithms still face various challenges [5]. Improving the speed and robustness of the tracking algorithm has been the focus of many scholars. Henrique et al. [6] proposed the circulant structure of tracking-by-detection with kernels (CSK) algorithm, which applies the cyclic matrix theory and the kernel method to correlation filter tracking. The CSK algorithm greatly improves the operation speed of the tracking algorithm, but the robustness of the algorithm is poor because it uses only grayscale features. Henrique et al. [7] proposed a kernel correlation filter (KCF) tracking algorithm based on multiple channel gradient direction histograms. The algorithm is an improvement on CSK, and the robustness has been greatly improved. In order to reduce cumulative error in the tracking process, Hare et al. [8] proposed an adaptive tracking framework based on structured output prediction to improve the robustness of the algorithm. However, the object's representation of the algorithm does not combine multiple features, which leads to tracking failure in more complex scenes. The Staple algorithm proposed by Bertinetto et al. [9] combines the discriminative scale space tracker with color histogram tracking, which improves the adaptability of the algorithm to object deformation, but it does not perform well in scenes with illumination variations. Although the real-time performance of the methods described above is good, their artificial design features mean that they often fail to fully characterize the essential attributes of the object, which results in the algorithm performing well only in a specific scene such as occlusion, illumination variations, and motion blurring, and exhibiting poor tracking performance in complex environments [10].

Recent years have seen improvement in the ability of deep learning methods to provide accurate recognition and prediction [11–14]. Object tracking technology has made a breakthrough with the latest advances in deep learning [15]. Li et al. [16] applied online convolution neural networks to the task of object tracking. A small-scale convolutional neural network was designed for training and updating object samples stored online. However, this method trains the network online, which means the network cannot be fully trained, resulting in low detection and tracking accuracy. Wang et al. [17] studied the characteristics of the convolution neural network model. By analyzing the image features output from each layer of the network model, a feature selection network is constructed for matching and tracking. However, due to the large scale of the network model, it is impossible to achieve real-time tracking on the embedded platform. Gao et al. [18] proposed an update-pacing frame work based on the ensemble post-processing strategy to mitigate the model drifting of discriminative trackers. The framework initializes a set of trackers to update model in different intervals and selects the tracker with smallest deviation score as the robust tracker for the later tracking. However, the MTS framework performs not so well in real-time tracking when the number of trackers increases.

With the increasing application of computer vision technology in driverless, robot, and other mobile devices, it is urgent to design a hardware acceleration method with excellent performance on embedded devices. Because the target detection algorithm needs to complete a large number of complex mathematical calculations, it puts forward higher requirements for the performance of embedded hardware, algorithm selection and algorithm based improvement. Due to the target detection algorithm needs to complete a large number of complex mathematical calculations, it puts forward higher requirements

for the performance of embedded hardware, algorithm selection, and algorithm-based improvement. At present, convolutional neural network is usually deployed in embedded system to quantify the weight or activation value, that is, to convert the data from 32-bit floating-point type to low-integer type, such as binary neural network(BNN), ternary weight network (TWN), and XNORNet. However, there are still some shortcomings in the current quantization methods in the trade-off between accuracy and calculation efficiency. Many quantization methods compress the network in different degrees and save storage resources, but they can not effectively improve the calculation efficiency in the hardware platform. In 2017, Xilinx proposed to quantify the weight of convolutional neural network from 32-bit to fixed-point 8-bit, adopted the method of software and hardware co-design, and realized the hardware acceleration of the model by FPGA, which met the real-time requirements of convolutional neural network in embedded system.

The main research content of this paper is to implement target detection and tracking in embedded hardware and software system. The task of target detection and tracking requires the algorithm to be real-time and robust. In order to solve the problems of poor real-time tracking performance of convolutional neural network and low robustness of tracking algorithm in complex scenes, a fast and accurate video real-time detection and tracking algorithm for embedded system is proposed. It is a tracking algorithm with object detection, which is based on a deep convolutional neural network single-shot multibox detector (SSD) [19] model and kernelized correlation filters(KCF) object tracking algorithm. In order to improve the real-time performance of the algorithm in the embedded system, the SSD model is quantized and compressed in Xilinx DNNDK (Deep Neural Network Development Kit) development environment, and the compressed model is deployed to the embedded system by the method of software and hardware co-design. The main work of this paper is as follows:

(1) To achieve higher robustness in complex scenes, a deep learning method is applied to object detection. Due to the high detection accuracy of the SSD model, that model is used to locate the object. It is important to mention that we are not proposing a new or improved version of SSD, but rather a method for the hardware-software co-design of embedded systems based on System on Chip (SoC).

(2) To achieve higher speeds, a KCF is applied to object tracking. In complex scenes, such as fast movement, camera shake, and occlusion, the KCF algorithm tends to track failures. After the failure, the KCF model is updated with the wrong object samples, which will contaminate the model, causing it to be unable to continue tracking the object [20]. This paper proposes a validity detection mechanism of tracking results to judge whether the tracking fails or not, so as to decide whether to update the model.

(3) Since the missed rate of the SSD model in the scenes of blurred motion and illumination variation is high, this paper introduces a strategy to reduce the missed rate. When missed detection occurs in the process of object detection, the object position in the current frame is predicted according to the position of the object in the previous two frames, so as to reduce the missed rate.

(4) In order to improve the real-time performance of the algorithm in the embedded hardware and software system, the SSD model is quantized as an 8-bit fixed-point model, the algorithm was partitioned through the way of hardware and software co-design, and part of the tasks was completed by ARM and FPGA, respectively. In this way, the advantages of ARM and FPGA are fully exploited, and the real-time performance is achieved without loss of accuracy.

The rest of this paper is organized as follows. Section 2 introduces the real-time detection-tracking algorithm for embedded systems, including a validity detection mechanism of tracking results, and the strategy to reduce the missed rate. In Section 3, the method for hardware and software co-design of embedded systems based on SoC is described. Section 4 compares the results with other representative methods. Finally, the conclusion is given in Section 5.

2 Proposed method

In this section, we will introduce our algorithm for real-time object detection and tracking in embedded systems. In order to achieve an adequate real-time performance, the algorithm obtains the object box information by the SSD model only on the key frame. The object box information includes the location and size of the object. KCF target tracking algorithm separates target and background through discriminant framework, so as to achieve the goal of target tracking. The KCF model is trained through samples, which are obtained by cyclic shifting from the region inside the object box. In order to avoid the contamination of the KCF model caused by tracking failure, this paper introduces the validity detection mechanism of tracking results to evaluate whether tracking has failed or not and then choose to update the model or retrain it through SSD object detection results. A strategy is introduced to reduce the missed rate of the SSD model in motion-blurred and illumination-variation scenes.

The overall flow of the algorithm is shown in Fig. 1. The first step is to run either the SSD object detection algorithm or the KCF object tracking algorithm on the frame i (image I_i):

$$S(I_i) = \begin{cases} SSD(I_i), & i \bmod N = 0 \text{ or } fr = 1 \\ KCF(I_i), & \text{otherwise} \end{cases} \tag{1}$$

where $S(I_i)$ indicates the detection or tracking method for I_i, $SSD(I_i)$ is the SSD object detection method, $KCF(I_i)$ is the KCF tracking method. N is a constant of value 20 in this paper. fr is a flag of value 1 when the validity detection mechanism of tracking results fails.

For SSD object detection or KCF object tracking, the tracking or detection can be expressed as:

$$\begin{cases} L_s(l_i, c_i, r_i, n_i) = S(I_i), & S(I_i) = SSD(I_i) \text{ or } F(r_i, r_{i-1}) = 0 \\ L_K(r_i) = S(I_i), & \text{otherwise} \end{cases} \tag{2}$$

where $L_s(l_i, c_i, r_i, n_i)$ is the result of SSD object detection, l_i represents the object category, c_i is the confidence of the category, r_i is the object box of the detection result, and n_i is the object number. $F(r_i, r_{i-1})$ is the result of validity detection mechanism of tracking results. The calculation method is given in Section 2.1. $L_K(r_i)$ is the result of the KCF tracking.

If n_i is 0—that is, no object is detected—the strategy of reducing the missed detection is used to reduce the missed rate. The calculation method is given in Section 2.2. Otherwise, based on the image blocks contained in r_i, the samples needed for KCF training can be obtained by cyclic shifting, so as to train the object initial position model for subsequent object tracking.

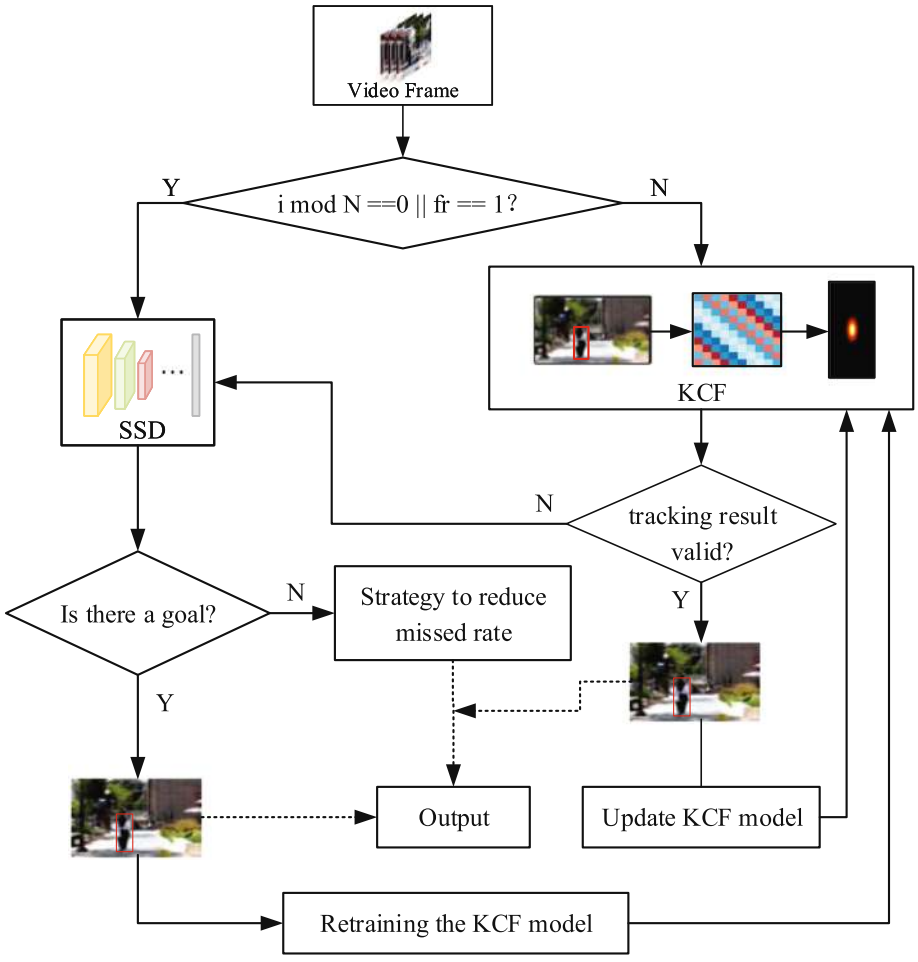

Fig. 1 Flowchart of the proposed method

2.1 Validity detection mechanism of tracking results

The KCF tracking algorithm in this paper is updated by linear interpolation as shown in Equation 3 [7]:

$$\begin{cases} \alpha_t = (1-\eta) \times \alpha_{t-1} + \eta \times \alpha_t{}' \\ x_t = (1-\eta) \times x_{t-1} + \eta \times x_t{}' \end{cases} \tag{3}$$

where η is an interpolation coefficient, which characterizes the learning ability of the model for new image frames, α_t is the classifier model, and x_t is the object appearance template. It can be seen that the KCF algorithm does not consider whether the prediction result of the current frame is suitable for updating the model. When the tracking results deviate from the real object due to occlusion, motion blur, illumination variation, and other problems, Equation 3 will incorporate the wrong object information into the model, which will gradually contaminate the tracking model and eventually lead to the failure of subsequent object tracking tasks. In order to avoid inaccurate tracking caused by model contamination, it is necessary to judge whether tracking failure has occurred in time. In the process of tracking, the differences between the object information of adjacent frames can be expressed by the correlation of the object area. When the tracking is

successful, the difference between the object regions of adjacent frames is very small and the correlation is very high. When the tracking fails, the object regions of adjacent frames will change greatly, and the correlation will also change significantly. Therefore, this paper adopts the correlation of frame objects to judge whether or not tracking fails.

Considering that the application object of this algorithm is an embedded system, in order to improve the real-time performance of the algorithm, we only use low-frequency information to calculate the correlation. The information in the image includes high-frequency and low-frequency components: high-frequency components describe specific details, while low-frequency components describe a wide range of information. Figure 2a is a frame image randomly selected from the BlurBody video sequence in the OTB-100(Object Tracking Benchmark) [5] dataset, and Fig. 2b is a matrix diagram of the discrete cosine transform coefficients of the image. From Fig. 2b, it can be seen that the image energy in natural scenes is concentrated in the low-frequency region. In addition, some conditions such as camera shake and fast motion of the object may also cause motion blur, which may result in insufficient high-frequency information. Therefore, the high-frequency information is not reliable in judging the correlation of the object area.

In this paper, a perceptual hash algorithm [21] is proposed to quickly calculate the hash distance between the object area of the current frame and the previous frame. This process uses only low-frequency information. The hash distance is the basis for judging whether the tracking fails, as shown in Equation 4.

$$F(r_i, r_{i-1}) = \begin{cases} 1 & pd_{i,i-1} \leq H_{th} \\ 0 & pd_{i,i-1} > H_{th} \end{cases} \tag{4}$$

where $F(r_i, r_{i-1})$ indicates whether the frame i fails to track, which is determined by the object area of the frame $i-1$ in the video sequence, the values of 1 and 0 representing tracking success and tracking failure, respectively; $pd_{i,i-1}$ is the hash distance between the object area of frame i and frame $i-1$; and H_{th} is the hash distance threshold.

Taking the BlurBody video sequence in the OTB-100 dataset as the test object, the hash distance $pd_{i,i-1}$ between the real object area of each frame and the previous frame is calculated, as shown in Fig. 3.

It can be seen from Fig. 3 that the hash distance of the object area is usually less than 15; for video frames with $pd_{i,i-1}$ greater than 15, there are often obvious blurring and camera shakes. At this time, there are significant deviations in the tracking results of the KCF algorithm.

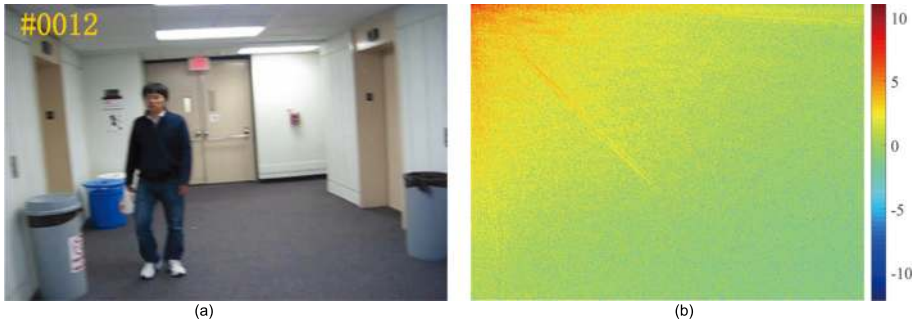

Fig. 2 Video frame and its matrix of discrete cosine transform coefficients. **a** Frame 12 of the BlurBody video sequence. **b** Matrix of discrete cosine transform coefficient

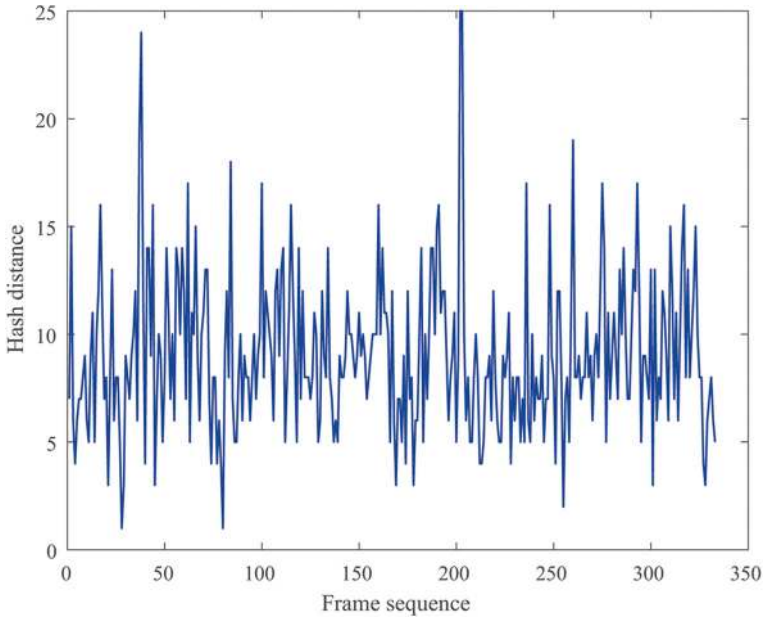

Fig. 3 Hash distance between target regions of adjacent frames in BlurBody video sequence

Figure 4 shows the BlurBody video sequence tested by the KCF algorithm. The tracking results of frames 43, 107, and 160 are compared with the real position of the object, and the tracking result hash distances $pd_{43,42}$, $pd_{107,106}$, and $pd_{160,159}$ are respectively 9, 22, and 15. The hash distance $pd_{i,i-1}$ of frame 43 is lower, and the tracking result is more accurate; the hash distance $pd_{i,i-1}$ of frame 107 is higher, and the tracking result has obviously deviated from the true position of the object. It can be seen that the hash distance $pd_{i,i-1}$ can well reflect the validity of the tracking result.

2.2 Strategy to reduce missed rate

There is less information on the appearance of images in the motion blur and dark scenes [22]. In addition, the SSD model detects the current frame separately—it does not consider the correlation of adjacent frames, so the missed rate is high in the above scenes. In this paper, image enhancement is proposed to obtain more detailed image information, and then the improved KCF algorithm is used to track the object in order to reduce the missed rate.

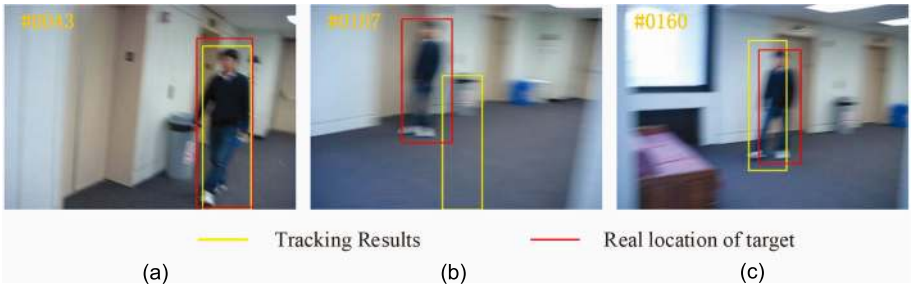

Fig. 4 Tracking results compared with real positions. **a** Comparison result of frame 43. **b** Comparison result of frame 107. **c** Comparison result of frame 160

We are faced with the situation that the SSD model cannot detect the object when the image is blurred or dark, as shown in Fig. 5. The essence of image blurring or darkening is that the image is subjected to average or integral operation, so the image can be inversely calculated to highlight the details of the image. In this paper, the Laplacian differential operator is proposed to sharpen the image to obtain more detailed information.

The enhanced image is tracked by an improved KCF algorithm with color features. In the KCF tracking algorithm, the object feature information is described by the histograms of oriented gradients [23]. However, in images involving blurring or illumination variation, the edge information of the object is often not obvious. This paper proposes to extract object information by combining color features using the Lab color feature. The strong expressive ability of the Lab color space allows for a better description of the appearance of the object.

In the KCF tracking algorithm, for the case of extracting multi-channel features of an image as input, it is assumed that the describing vector of each channel feature of an image is $\mathbf{x} = [x_1, x_2, \cdots x_C]$, and the output formula of Gauss kernel in reference [7]:

$$\mathbf{k}^{xx'} = \exp\left\{-\frac{1}{\sigma^2}\left\{\|\mathbf{x}\|^2 + \left\|\mathbf{x}'\right\|^2 - 2\mathcal{F}^{-1}\left[F(\mathbf{x}) \odot F^*\left(\mathbf{x}'\right)\right]\right\}\right\} \tag{5}$$

could be rewritten as:

$$\mathbf{k}^{xx'} = \exp\left\{-\frac{1}{\sigma^2}\left[\|\mathbf{x}\|^2 + \left\|\mathbf{x}'\right\|^2 - 2\mathcal{F}^{-1}\left(\sum_{c=1}^{C} F_C\right)\right]\right\} \tag{6}$$

where $F_C = F(\mathbf{x}_C) \odot F^*\left(\mathbf{x}'_C\right)$.

Based on Equation 6, the object is described by the histogram of oriented gradients feature of the 31-channel feature. In the strategy to reduce missed rate, Laplacian sharpening is first applied to the previous two frames. Then, KCF tracking model with the Lab color feature is trained by the object in the sharpened image. Next, the object position of the current frame is predicted by the trained model. Finally, the tracking result is checked by the method described in Section 2.1. If the tracking is successful, the predicted object will be given as the result. Otherwise, the object in the next frame will continue to be detected by the SSD model. The algorithm flow is shown in Fig. 6.

To verify the feasibility of the algorithm in this section, two motion-blurred video sequences, BlurBody and BlurOwl, and two illumination-varying video sequences, Human 9 and Singer 2, were selected from the OTB-100 dataset for the following comparison experiments:

Fig. 5 Examples of complex scenes. **a** Motion blur. **b** Light tends to be dark in the case of illumination variation

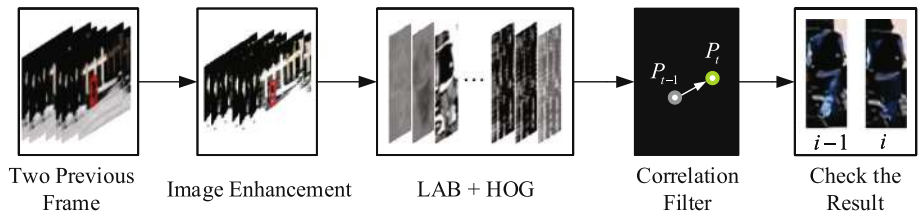

Fig. 6 Flow charts of the strategy to reduce missed rate

Experiment 1 The object in the four video sequences is tracked by an unimproved KCF algorithm.

Experiment 2 Clear frame sequences, or frame sequences having no significant illumination variations, were tracked by the unimproved KCF algorithm. Only the partial frame sequences of motion blur or illumination variations were tracked by the algorithm described in this section.

The tracking results of Experiment 1 and Experiment 2 were evaluated by two indexes: precision rate (PR) and success rate (SR). The PR and SR of experiments 1 and 2 are shown in Fig. 7. It can be seen from the figure that for the video sequences of motion blur and illumination variation, the improved KCF tracking algorithm exhibits a significantly higher PR and SR than the unimproved algorithm.

3 Hardware and software co-design

The algorithm in this paper is implemented by Zynq UltraScale+ MPSoC (multiprocessor system-on-chip) [24, 25]. The real-time object detection and tracking algorithm based on the SSD model and KCF tracking is implemented by embedded software and hardware cooperation: the algorithm module, with simple operation, a mass of judgment statements, and pointer operation, is processed by the processing system (PS); the part with great influence on the speed performance of the algorithm and high degree of parallelism is implemented by programmable logic (PL), which is composed of FPGA. The hardware and software partition of the system is shown in Fig. 8. The convolution and pooling layers in SSD model are implemented by the hardware in PL, while the softmax layer is implemented by PS because it involves floating-point operation. In addition, other functions are implemented by PS, such as non-maximum supression, mapping the operation results

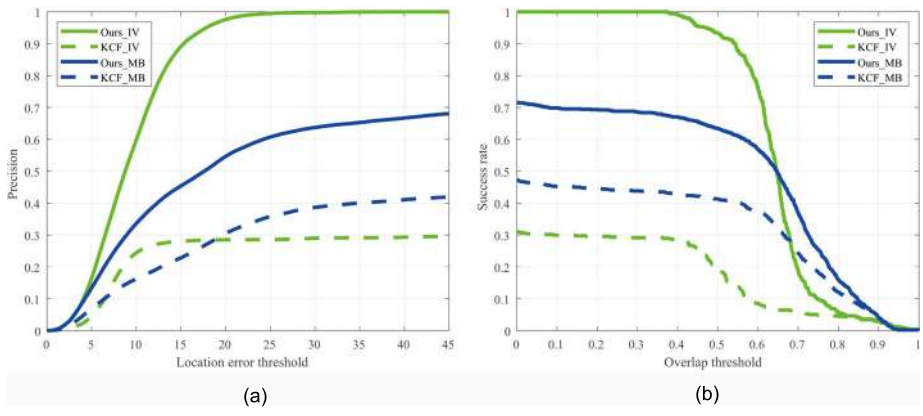

Fig. 7 Test results of comparative experiments. **a** Precision plot. **b** Success plot

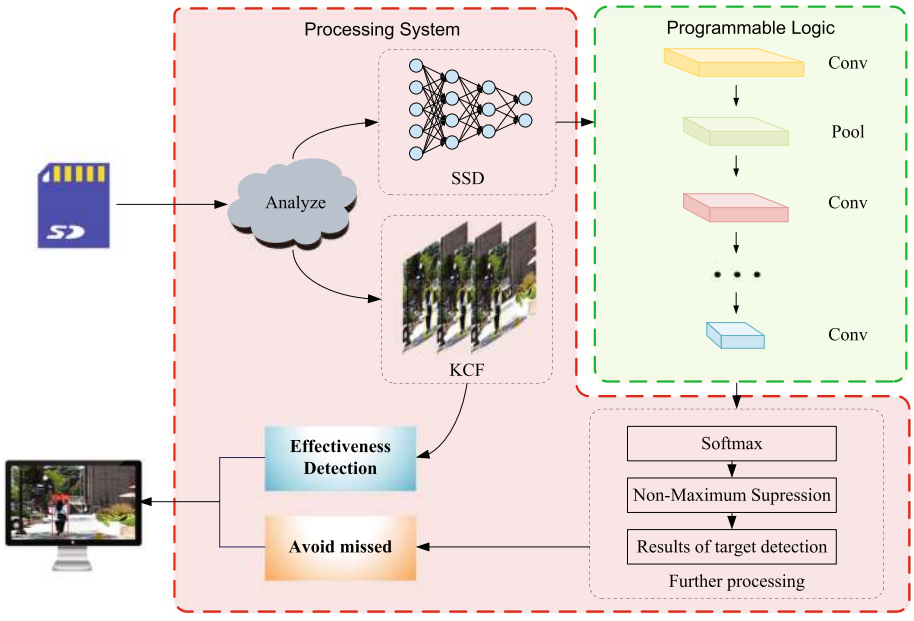

Fig. 8 Software and hardware partition of the system

to the image, KCF tracking algorithm, validity detection mechanism of tracking results, and strategy to reduce missed rate.

The computing capability and memory bandwidth of embedded systems are limited; in addition, the weight parameters of deep neural network model often have a lot of redundancy. When the system on chip is implemented, there will be bottlenecks in computing and storage resources, but also a lot of power consumption. In this paper, we use DNNDK (Deep Neural Network Development Kit) [26] to compress the 32-bit floating-point weight into 8-bit fixed-point weight,and then compile it into ARM-executable instruction stream. In the embedded system, PS obtains the instruction from the off-chip memory and transmits it to the on-chip memory of the PL through the bus. The instruction scheduler of the PL obtains the instruction to control the operation of the computing engine [27, 28], as shown in Fig. 9. On-chip memory is used to cache input and output data as well as temporary data in the operation process, so as to achieve high throughput. On-chip memory is used to cache input and output data as well as temporary data in the operation process, so as to achieve high throughput. The deep pipeline design is adopted in the computing engine, which makes full use of the parallel processing function of FPGA, so the computing speed is improved. Among them, the processing elements of convolutional computing engine make full use of the fine-grained blocks in PL, such as multiplier, adder and so on, which makes it possible to efficiently complete the computation in convolutional neural network.

In this paper, the tasks of reading video frames, running detection and tracking algorithms, and displaying video frames are implemented by ARM of the PS. If the single thread mode is adopted, the image is read first, then the target is detected and tracked, and finally the image is displayed, which will make the utilization of CPU and FPGA relatively low, and the real-time performance will be affected. Since the reading of video frame is mainly file I/O operation, the detection and tracking algorithm is mainly CPU and

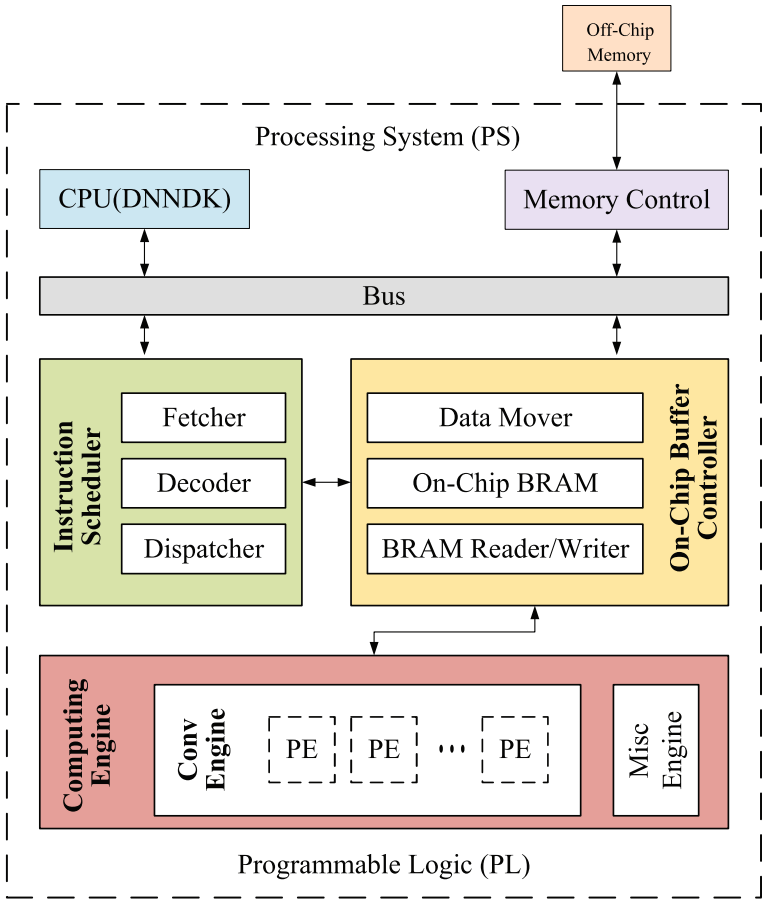

Fig. 9 Hardware architecture of software and hardware co-design

FPGA calculation, and the video image display is mainly Ethernet transmission (displayed after being transmitted to the computer through X11 protocol). It can be seen that these three steps occupy different system resources, so using multithreading to execute the above three steps simultaneously can make full use of CPU resources, thus significantly improving the operation efficiency, as shown in Fig. 10.

4 Results and discussion

4.1 Overall performance comparison of algorithms

In this section, we compare the object tracking performance of the proposed algorithm with four other algorithms that have better real-time and robustness: KCF, Struck, CSK, and Staple. In order to evaluate the performance of the proposed algorithm, the tracking algorithm, the SSD object detection model, and the algorithm itself are tested on the

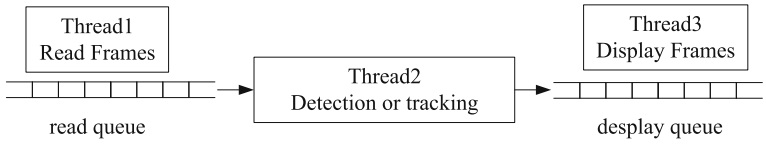

Fig. 10 Multithread concurrent execution of PS

OTB-100 dataset. To ensure the objectivity and fairness of the experimental results, the SSD model in this algorithm is trained by the open datasets VOC2007 [29] , VOC2012 [30], and Wider Face [31]. Since these three training datasets differ greatly from the object categories of the OTB-100 test dataset, 14 video sequences are tested, which are BlurBody, CarScale, Couple, Girl, Gym, Human2, Human6, Human7, Human8, Human9, Liquor, Man, Trellis, and Woman.

We chose the one pass evaluation method [5] to test the KCF, Struck, CSK, and Staple tracking algorithms for artificially setting the first frame object position. The SSD object detection model and the algorithm in this paper do not provide the initial object position, and the object position is automatically detected by the model. Table 1 shows the results of the comparison of these algorithms, measured by PR, SR , and tracking speed (in frames per second).

The precision plot and success plot are drawn to prove the feasibility and effectiveness of the proposed algorithm, as shown in Fig. 11.

It can be seen from Table 1 and Fig. 11 that the SR of the proposed algorithm is slightly higher than the SSD object detection model, and the PR is nearly 10% higher. This is attributed to the fact that the algorithm remedies the shortcomings of the SSD model by using the strategy of reducing missed detection in the scenes involving motion blur and illumination variation, which makes it possible to track frames that the SSD model cannot detect. Compared with Struck, CSK, and KCF, the PR and SR of the proposed algorithm are much higher. This is due to the validity detection mechanism of tracking results, which can detect tracking failure in time and immediately start the SSD model to re-detect the object.

When testing processing speed, KCF, Struck, CSK, and Staple are tested directly on a PC. The experimental environment is the Windows operating system with an Intel® Core i7-7700K 4.2-GHz processor with 16 GB of memory. The experimental software is MATLAB R2018b. The SSD detection model and the proposed algorithm are tested on a ZCU104 board [32]. Taking advantage of the fast running speed of KCF, it can achieve 36.2 frames per second on the ZCU104 board, which meets the real-time requirements.

4.1.1 Experiment after adding training dataset

As can be seen from Table 1 and Fig. 11, the PR and SR of the proposed algorithm are not higher than those of Staple. This is because the SSD model is needed to detect the object position when the current frame is the initial frame or in the tracking failure scenes. However, the training datasets VOC2007 and VOC2012 of the SSD model are quite different from the objects in the test video sequence: the "person" category in the test dataset is mostly the image under the road monitor perspective, while the "person" image in the training dataset is all the image from the horizontal perspective. In addition, the video sequence of the "bottle" category in the test dataset is very long, whereas the images in the training dataset are too few in number.

Table 1 The contrast of tracking accuracy and speed result

Method	SSD	Struck	CSK	Staple	KCF	Ours
PR(%)	74.51	49.92	35.73	93.62	69.71	83.27
SR(%)	75.12	45.26	28.69	74.14	58.94	79.35
FPS(frame/s)	**13.73**	16	791	85	242	**36.2**

The values in bold are tested in the ZCU104 board

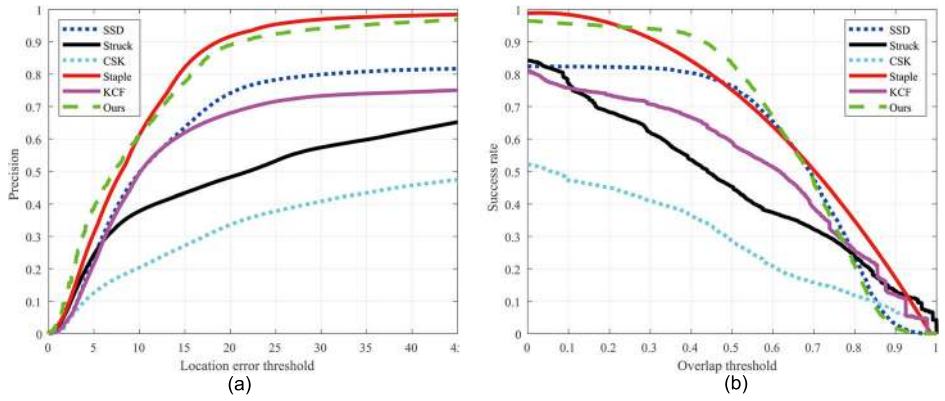

Fig. 11 One pass evaluation. **a** Precision plot. **b** Success plot

The significant difference between the training dataset and the test dataset leads to the low accuracy of the SSD model (PR and SR are 74.51% and 75.12%, respectively), which reduces the accuracy of the proposed algorithm. To verify this point, another experiment was conducted: about 200 bottle images and 200 pedestrian images from the perspective of the road monitor were added to the training dataset. It is important to note that all the images added to the training dataset were not included in the test dataset. After retraining the SSD model, the test results of the contrast experiment are shown in Table 2. It can be seen that with the increase in the number of images in the training dataset, the accuracy of the SSD model has been improved slightly, while the accuracy of the algorithm in this paper has been greatly improved. At this time, the PR of the algorithm is approximately the same as that of Staple, but its SR is about 10% higher than that of Staple.

4.2 Experiments with specific attributes

Object tracking tests were carried out on the video sequences with the attributes of occlusion, motion blur, and illumination variation. The accuracy and robustness of the algorithm in these challenging conditions are compared using the SR. The results are shown in Table 3.

It can be seen from Table 3 that for occlusion, the SR of the algorithm and the SSD model are higher than those of other tracking algorithms. In other tracking algorithms, when the object is occluded, the algorithm not only easily loses track of the object, but also often fails to relocate the object in order to continue. However, the algorithm in this paper can relocate the object after tracking failure and thus continue to track it through the SSD model.

For motion blurring, the SR of the proposed algorithm and the Staple algorithm is higher than the other algorithms. This is attributed to the introduction of color features into the strategy of reducing missed detection in this paper.

Table 2 Test results after adding training data

Method	SSD		Ours	
	Test results	Increased	Test results	Increased
PR (%)	76.92	2.41	91.19	7.92
SR (%)	76.23	1.11	84.79	5.44

Table 3 Tracking result on videos with sequence attributes

Method	Success rate (%)		
	Occlusion	Motion blur	Illumination variation
SSD	84.30	77.54	77.21
Struck	46.87	32.63	19.61
CSK	23.72	47.60	23.27
Staple	81.91	99.70	74.08
KCF	56.99	72.75	31.33
Ours	84.47	92.21	86.53

For illumination variation, the SSD model exhibits high accuracy, because the brightness and contrast of the training dataset have been changed randomly in order to improve generalization in training the SSD model. In illumination-variation scenes, when the light is dark, the SR has been greatly improved compared with the SSD model due to introducing color features into the strategy of reducing missed detection.

5 Conclusions

This paper has presented a real-time object detection-tracking algorithm for embedded platforms. This algorithm combines the object detection model of SSD in deep convolution networks and the KCF tracking algorithm. The SSD model was accelerated by using field-programmable gate arrays, which satisfies the real-time performance of the algorithm in embedded platforms. To solve the problem of the traditional KCF algorithm not being able to robustly track for a long time, and specifically the problem of contamination after it fails to track in complex scenes, the validity detection mechanism of the tracking results was proposed. To solve the problem of the SSD model's high missed rate under the conditions of motion blur and illumination variation, a strategy to reduce the missed detection was proposed by introducing color features. The results of the experiments show that the overall PR of the proposed algorithm reaches 91.19%, the SR reaches 84.79%, and the frame rate reaches 36.2 frames per second. In the specific cases of occlusion, motion blurring, and illumination-variation attributes, the proposed algorithm has higher accuracy and robustness than other tracking algorithms. The focus of future research should be on improving tracking performance by tracking with deep features and improving hardware implementation.

Acknowledgements
The authors would like to thank the editor and anonymous reviewers for their helpful comments and valuable suggestions.

Authors' contributions
All authors take part in the discussion of the work described in this paper. QJ proposed the framework of this work. CD carried out the whole experiments and drafted the manuscript. CH offered useful suggestions and helped to modify the manuscript. XL analyzed the data. All authors read and approved the final manuscript.

Authors' information

1. Qingbo Ji received her B.S, M.S, and PhD degrees in College of Information and Communication Engineering from Harbin Engineering University, Heilongjiang, China, in 1998, 2004, 2008, respectively. She is currently an associate professor with College of Information and Communication Engineering, Harbin Engineering University. Her major research interests include the image processing, image recognition, and deep learning and embedded systems.
2. Chong Dai received his B.S degree in College of Electronic and Information Engineering from Heilongjiang University of Science and Technology, Heilongjiang, China, in 2017. And now, he is a postgraduate in College of Information and Communication Engineering at Harbin Engineering University.
3. Changbo Hou received the B.S and M.S degree in College of Information and Communication Engineering from Harbin Engineering University, Heilongjiang, China, in 2008 and 2011, respectively. He is currently a lecturer with the College of Information and Communication Engineering, Harbin Engineering University, where he is also a

Doctor in reading in the Key Laboratory of In-ber Integrated Optics, Ministry Education of China. His research interests include wideband signal processing, optical sensors, image processing, and deep learning.

4. Xun Li received his B.S degree in commutation engineering from Northeastern University, Qinhuangdao, China, in 2017. And now, he is a postgraduate in College of Information and Commutation Engineering at Harbin Engineering University.

Funding
The work is founded by the National Key Research and Development Program of China (No. 2018AAA0102700).

Availability of data and materials
We used publicly available data in order to illustrate and test our methods. The datasets used were acquired from [5] and from [29–31], which can be found in https://pjreddie.com/media/files/VOCtrainval_06-Nov-2007.tar, https://pjreddie.com/media/files/VOCtrainval_11-May-2012.tar, and http://shuoyang1213.me/WIDERFACE/, respectively.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 26 June 2019 Accepted: 18 May 2021
Published online: 16 June 2021

References

1. L. Hongmei, H. Lin, Z. Ruiqiang, L. Lei, W. Diangang, L. Jiazhou, in *2020 International Conference on Computer Engineering and Intelligent Control (ICCEIC)*, Object tracking in video sequence based on Kalman filter, (2020), pp. 106–110. https://doi.org/10.1109/ICCEIC51584.2020.00029
2. Y. Wang, W. Shi, S. Wu, Robust UAV-based tracking using hybrid classifiers. Mach. Vis. Appl. **30**(1), 125–137 (2019). https://doi.org/10.1007/s00138-018-0981-4
3. R. Iguernaissi, D. Merad, K. Aziz, P. Drap, People tracking in multi-camera systems: a review. Multimed. Tools Appl. **78**(8), 10773–10793 (2019). https://doi.org/10.1007/s11042-018-6638-5
4. H. Zhang, Z. Zhang, L. Zhang, Y. Yang, Q. Kang, D. Sun, Object tracking for a smart city using IoT and edge computing. Sensors. **19**(9), 1987 (2019). https://doi.org/10.3390/s19091987
5. Y. Wu, J. Lim, M.-H. Yang, Object tracking benchmark. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1834–1848 (2015). https://doi.org/10.1109/TPAMI.2014.2388226
6. J. F. Henriques, R. Caseiro, P. Martins, J. Batista, in *Computer Vision ? ECCV 2012. ECCV 2012. Lecture Notes in Computer Science, vol. 7575*. ed. by A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Exploiting the circulant structure of tracking-by-detection with kernels (Springer, Platz, 2012), pp. 702-715
7. J. F. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2015). https://doi.org/10.1109/TPAMI.2014.2345390
8. S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, P. H. S. Torr, Struck: structured output tracking with kernels. IEEE Trans. Pattern Anal. Mach. Intell. **38**(10), 2096–2109 (2016). https://doi.org/10.1109/TPAMI.2015.2509974
9. L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, P. H. S. Torr, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Staple: complementary learners for real-time tracking (IEEE, Seattle, p. 2016. https://doi.org/10.1109/CVPR.2016.156
10. Y. Chen, X. Yang, B. Zhong, S. Pan, D. Chen, H. Zhang, CNNTracker: online discriminative object tracking via deep convolutional neural network. Appl. Soft Comput. **77**, 1088–1098 (2016). https://doi.org/10.1016/j.asoc.2015.06.048
11. K. Zhang, Y. Guo, X. Wang, J. Yuan, Q. Ding, Multiple feature reweight DenseNet for image classification. IEEE Access. **6**, 9872–9880 (2019). https://doi.org/10.1109/ACCESS.2018.2890127
12. T. Kong, A. Yao, Y. Chen, F. Sun, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, HyperNet: towards accurate region proposal generation and joint object detection IEEE, Seattle, 2016), pp. 845-853. https://doi.org/10.1109/CVPR.2016.98
13. T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, S. Belongie, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Feature pyramid networks for object detection (IEEE, Honolulu, 2017), pp. 1-8. https://doi.org/10.1109/CVPR.2017.106
14. N. Bodla, B. Singh, R. Chellappa, L. S. Davis, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Soft-NMS - improving object detection with one line of code (IEEE, Venice, 2017), pp. 5562-5570. https://doi.org/10.1109/ICCV.2017.593
15. S. M. Marvasti-Zadeh, L. Cheng, H. Ghanei-Yakhdan, S. Kasaei, Deep learning for visual tracking: a comprehensive survey. IEEE Trans. Intell. Transp. Syst. https://doi.org/10.1109/TITS.2020.3046478
16. H. Li, Y. Li, F. Porikli, DeepTrack: learning discriminative feature representations online for robust visual tracking. IEEE Trans. Image Process. **25**(4), 1834–1848 (2016). https://doi.org/10.1109/TIP.2015.2510583
17. L. Wang, W. Ouyang, X. Wang, H. Lu, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Visual tracking with fully convolutional networks (IEEE, Santiago, 2015), pp. 3119-3127. https://doi.org/10.1109/ICCV.2015.357
18. Y. Gao, Z. Hu, H. W. F. Yeung, Y. Y. Chung, X. Tian, L. Lin, in *IEEE Transactions on Circuits and Systems for Video Technology, vol. 30*, Unifying temporal context and multi-feature with update-pacing framework for visual tracking, (2020), pp. 1078–1091. https://10.1109/TCSVT.2019.2902883
19. W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, in *Computer Vision ? ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol. 9905*. ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling, SSD: single shot multibox detector (Springer, Cham, 2016), pp. 21–37. https://doi.org/10.1007/978-3-319-46448-0_2

20. M. Jiang, J. Shen, J. Kong, H. Huo, Regularisation learning of correlation filters for robust visual tracking. IET Image Process. **12**(9), 1586–1594 (2018). https://doi.org/10.1049/iet-ipr.2017.1043
21. Y. Zhen, D.-Y. Yeung, Active hashing and its application to image and text retrieval. Data Min. Knowl. Disc. **26**(2), 255–274 (2013). https://doi.org/10.1007/s10618-012-0249-y
22. Z. Zhan, X. Yang, Y. Li, C. Pang, Video deblurring via motion compensation and adaptive information fusion. Neurocomputing. **341**, 88–98 (2019)
23. N. Dalal, B. Triggs, in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1.*, Histograms of oriented gradients for human detection (IEEE, San Diego, 2005), pp. 886-893
24. Xilinx, Zynq UltraScale+ MPSoC ZCU104 Evaluation Kit Quick Start Guide. Available: https://www.xilinx.com/support/documentation/boards_and_kits/zcu104/xtp482-zcu104-quickstart.pdf. Accessed May 2018
25. Xilinx, SDSoC environment user guide. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2019_1/ug1027-sdsoc-user-guide.pdf. Accessed May 2019
26. Xilinx, DNNDK user guide. Available: https://www.xilinx.com/support/documentation/user_guides/ug1327-dnndk-user-guide.pdf. Accessed Apr 2019
27. Xilinx, Xilinx AI SDK user guide. Available: https://www.xilinx.com/support/documentation/user_guides/ug1354-xilinx-ai-sdk.pdf. Accessed Apr 2019
28. Xilinx, Xilinx AI SDK programming guide. Available: https://www.xilinx.com/support/documentation/sw_manuals/vitis_ai/1_1/ug1355-xilinx-ai-sdk-programming-guide.pdf. Accessed Apr 2019
29. M. Everingham, G. L. Van, C. K. I. Williams, J. Winn, A. Zisserman, The Pascal Visual Object Classes (VOC) Challenge. Int. J. Comput. Vis. **88**(2), 303–338. https://doi.org/10.1007/s11263-009-0275-4
30. M. Everingham, S. M. A. Eslami, G. L. Van, C. K. I. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge: a retrospective. Int. J. Comput. Vis. **111**(1), 98–136 (2015). https://doi.org/10.1007/s11263-014-0733-5
31. S. Yang, P. Luo, C. C. Loy, X. Tanal, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, WIDER FACE: a face detection benchmark (IEEE, Seattle, p. 2016. https://doi.org/10.1109/CVPR.2016.596
32. Xilinx, ZCU104 board user. Available: https://www.xilinx.com/support/documentation/boards_and_kits/zcu104/ug1267-zcu104-eval-bd.pdf. Accessed Oct 2018

Publisher's Note