Object Recognition and Search Based on Machine Vision

Junjie Chen[*1], Yuan Li[*2]

[*1] Beijing Institute of Technology, Beijing 100081, China
E-mail: bit_chenjunjie@163.com
[*2] Beijing Institute of Technology, Beijing 100081, China
E-mail: liyuan@bit.edu.cn

Abstract. The loss of personal belongings often occurs in daily life. This paper studies the object recognition algorithm and its implementation on the mobile terminal. Two offline-based mobile terminal object recognition apps are developed, which can realize real-time object recognition without network, and has good privacy security. In this paper, the object recognition function is realized based on SIFT and YOLOX respectively. The transplantation of these two methods on the mobile terminal is realized and two apps are developed based on Android system. The developed android app was tested through some pictures collected in the surrounding living environment or downloaded from the network. The test results realized the function of real-time object detection, and achieved a prediction accuracy of more than 90% based on the deep learning method.

Keywords: Object recognition, Android system, SIFT, YOLOX, NCNN

1. INTRODUCTION

The loss of personal belongings is a very common phenomenon in real life. People often spend a lot of time and energy searching for belongings. It may cause a lot of inconvenience to people's work and life if important items are lost and cannot be found in time.

With the development of machine vision, object recognition has become one of the important research directions of it. Object recognition can be used to recognize and label objects in the environment, which can realize the function of object search. Since the 21st century, the mainstream method of object recognition has focused on image recognition. Some classic image feature extraction methods include HOG [1], LBP [2] and so on. At the same time, the development of machine learning technology provides a new implementation method for object recognition. At present, convolutional neural network (CNN) and its improved algorithm are most widely used in the field of object recognition.

It is found that the existing apps on the market are often implemented based on the online method, that is, data preprocessing is performed on the mobile terminal, and then the data is sent to the server, where the prediction is completed and the results are sent to the mobile terminal device. Li and Wei (2016) implemented logo recognition on mobile terminal [3]. Shao et al. (2020) also implemented intelligent identification system of agricultural pests for mobile terminals on this method [4]. This method has the following defects: firstly, it relies on the Internet and needs to run when a network connection is established; Secondly, it is restricted by the network transmission speed and communication delay, so it cannot carry out real-time detection; Finally, users need to upload the living environment image to the external server, which has hidden privacy security risks.

To solve the problem, this paper implements an offline-based mobile terminal item finder app, which has good real-time and privacy performance, and does not need to rely on the network.

In this paper, firstly, the SIFT is used to obtain the gray image by graying the target object image. On this basis, the feature points are detected and matched according to the feature point description vector, and filtered according to distance. Then, the image is corrected by perspective transformation. After that, Auto.js and OpenCV are implemented to develop an app based on SIFT. Based on the YOLOX network model, two lightweight network models (YOLOX-Tiny and YOLOX-Nano) are trained by using the data of 38 kinds of common objects in life. After that, the trained network model was transplanted to the Android project through the NCNN platform, and the app was developed on Android Studio. This deep learning method achieved an accuracy of more than 90%.

2. MACHINE VISION ALGORITHM

2.1. SIFT

SIFT (Scale-invariant Feature Transform) is an algorithm in the field of image processing. It was first published by David Lowe [5] in 1999. It has scale invariance and can detect keypoints in images. It is a local feature descriptor. SIFT uses Difference of Gaussian to detect potential extremum with scale invariance and rotation invariance in different scale spaces. Then, the stable extremum is selected as the keypoint by fitting the model that reflects the position and scale. The direction of keypoints is

calculated according to the gradient. Finally, the description of keypoints is established according to the local gradient of the image to realize feature matching.

2.1.1. Scale Space

Scale space was proposed by Iijima [6] in 1962. Its main idea is to obtain image information at different scales by performing scale transformation on the original image [7]. The main contour of these images is extracted and used as the feature vector to realize edge and corner detection and feature extraction at different resolutions.

When constructing the scale space, SIFT selects Gaussian kernel for filtering. According to the derivation of Babaud [8] and Lindeberg [9], Gaussian kernel is the only linear kernel which can realize the scale transformation and save the most detailed features. Gaussian filtering is used to simulate the feature representation in large-scale cases.

The scale space $L(x, y, \sigma)$ of an image is defined as the convolution operation between the original image $I(x, y)$ and a two-dimensional Gaussian function $G(x, y, \sigma)$ with variable scale [10].

$$G(\gamma_i, y_i, \sigma) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{(x - x_i)^2 + (y - y_i)^2}{2\sigma^2}\right) \quad (1)$$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (2)$$

σ is the scale space factor, which indicates the degree to which the image is smoothed. The larger the value, the larger the scale, and the smoother the image. The image can reflect the general features of the object at a larger scale and the detailed features at a smaller scale.

2.1.2. Difference of Gaussian

A pyramid model consisting of images of different sizes can be obtained by continuously reducing the order of the original image. On this basis, Gaussian filter is added to ensure the continuous change of scale. The images of each layer of the pyramid are smoothed at multiple scales [11]. Each layer of the pyramid is composed of multiple Gaussian blurred images. Each layers is called an Octave. The group contains multiple images, each of which is called an Interval.

The Difference of Gaussian (DoG) is defined as:

$$\begin{aligned} D(x, y, \sigma) &= [G(x, y, k\sigma) - G(x, y, \sigma)] * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (3)$$

The two adjacent upper and lower images in each group of Gaussian pyramid are subtracted to obtain the Gaussian difference pyramid for extremum detection.

2.1.3. Keypoint Localization

There is a certain deviation between discrete extremum and continuous extremum detected by DoG. By fitting the DoG function with Taylor expansion, the position and scale of keypoints can be accurately determined, which makes the keypoints more stable.

The principal curvature of the DoG function is obtained from the Hessian matrix at the keypoints. Unstable edge points can be eliminated according to the magnitude of the principal curvature.

2.1.4. Orientation Assignment

In the neighborhood space of the keypoints, the gradient information at the keypoints is obtained by solving the gradient amplitude and direction.

Within 360°, the direction range is divided every 10°. The histogram is used to count the direction range of the keypoints. The peak direction of the histogram represents the main direction of the keypoints [12].

2.1.5. Keypoint Descriptor

Through the previous method, the position, scale and direction information of keypoints have been extracted. An abstract description is established for each keypoint, that is, the keypoint information is abstractly described through a set of vectors, which is called a keypoint descriptor. Descriptors also need to meet the invariance requirements such as scale invariance and translation invariance.

2.2. YOLOX

YOLOX is the latest generation detector of YOLO (You Only Look Once) series. It is based on YOLOv3-SPP and consists of CSPDarkNet, FPN and YOLOX Head. CSPDarkNet is the backbone feature extraction network of YOLOX, which outputs three effective feature layers. FPN is YOLOX's enhanced feature extraction network for feature fusion [13]. Yolox Head is the classifier and regressor of YOLOX.

Compared with the previous YOLO series, YOLOX mainly has the following improvements:

2.2.1. Anchor Free

Previously, the YOLO series used Anchor-based method, that is, for the objects in the picture, multiple anchors are used to return the position of the objects on the feature map. YOLOX adopts Anchor-free method, which regresses the position of anchor through a reference point. The reference point is on the feature map. Due to down sampling, this point is an anchor on the original drawing. Therefore, compared with Anchor-based method, Anchor-free eliminates the parameter adjustment pressure of manually set anchor, and the model post-processing is simpler and more efficient.

2.2.2. Decoupled Head

In versions before YOLOX, Coupled Head is used for detection. Classification, regression and objective scoring tasks are all detected through the head. YOLOX adopts Decoupled Head. There are two branches in the detection head. One branch is responsible for Cls, and the other branch divides into two small branches responsible for Reg and IoU respectively. This structure helps to improve the detection performance.

2.2.3. Strong Data Augmentation

YOLOX combines Mosaic and Mixup to improve performance. Mosaic achieves data augmentation by splicing four images together and training them as new

images. Mixup achieves data augmentation by aliasing two images and training them as new images. It is worth noting that in order to make the training pictures closer to the real scene, Mosaic and Mixup will be closed in the last 15 epoches during the training.

2.2.4. Multi Positives

Previously, Anchor-free would select only one positive sample for each central position. However, many high-quality predictions are not included in the calculation of loss, which is not conducive to training. In order to increase the number of positive samples required for each Ground Truth, Multi Positions is introduced. It takes a center 3*3 area as a positive sample instead of a point, also known as "Center Sampling", which can be used to build a Target.

2.2.5. SimOTA

In order to solve the assignment problem of fuzzy Anchor, OTA is introduced to transform the problem into label assignment task. However, due to its high computing cost, YOLOX proposed a simplified computing method called SimOTA.

SimOTA first uses the central a priori to determine the candidate regions of positive samples, and then calculates the loss of each sample to each Ground Truth. Then it uses the predicted samples of each Ground Truth to determine the number of positive samples (Dynamic k) it needs to be allocated. The first Dynamic k samples with the minimum loss for each Ground Truth are taken as positive samples. Finally, considering the global information, the case that the same sample is allocated to multiple Ground Truth positive samples is manually removed.

3. ANDROID SYSTEM

Android is an open source operating system developed by Google. It is based on the Linux kernel and is mainly used to support mobile devices.

3.1. Android System Structure

Android system adopts a layered architecture, divided into five layers: Application, Application Framework, Libraries, Android Runtime and Linux Kernel.

3.2. Android Application Component

There are some basic components of Android development:

- Activity: a single screen with a user interface which is used for interaction
- Service: a component that performs long-running operations in the background
- View: a component for building user interface elements
- Intent: the messaging object responsible for the launch of the component in the application

3.3. JNI

JNI is the abbreviation of Java Native Interface. Android's Java code and C/C++ code can call each other through JNI.

Generally, Java calls C/C++ code, and C/C++ code has to be compiled into dynamic link library (.dll/.so file). Java's Class Method can pass parameters to native C/C++ code and call Function through JNI interface, and then return the result to Class Method.

3.4. NDK

NDK (Native Development Kit) is a toolkit for Android development. It is mainly used for C/C++ development, which can help developers quickly develop C/C++ dynamic libraries and automatically package .so files and Java applications into apk [14].

After the C/C++ framework is compiled into library files by NDK, Java applications can call Native Libs through JNI interface to achieve the call of native C/C++ code.

4. DESIGN AND IMPLEMENTATION

4.1. Object Recognition App Based on SIFT

4.1.1. Development Environment

This project is based on Auto.js, version 9.0.10. Auto.js comes with OpenCV, version 4.5.1. The Android version is Android 12. The programming language is JavaScript.

4.1.2. Auto.js

Auto.js is an integrated development environment for writing and running JavaScript code on Android and Harmony platforms. It includes code completion editor, one-step debugging and graphical design. It can be built as an independent APK application or connected to a computer for development. Based on Auto.js can create automation tools, efficiency tools, beautiful interfaces and small applications. Auto.js has complete documents and examples, rich APIs, enhanced encryption, active updates, and uses JavaScript to connect Java, Android, and Node.js ecology.

4.1.3. OpenCV

OpenCV is an open source computer vision library, which is used for high-performance computing and emphasizes the development of real-time applications. It is based on C++, and the developer has deeply optimized it, making it have a strong advantage of multi-threaded processing.

OpenCV provides developers and researchers with an easy-to-use computer vision interface to help people quickly build sophisticated visual applications. OpenCV library contains more than 500 functions derived from various fields of computer vision, including industrial product quality inspection, medical image processing, security, interactive operation, camera calibration, binocular vision and robotics [15]. At the same time, because machine learning methods are often used in computer vision, OpenCV also contains a complete and universal machine learning library (ML module). This

sub library focuses on statistical pattern recognition and clustering. The ML module plays an important role in computer vision, and the library is generic enough to be used for any machine learning problem.

4.1.4. Algorithm Implementation

The implementation process is shown in **Fig. 1**. First, the target object image and database image are read in the album. Then OpenCV is called, and they are converted to grayscale images through Imgproc.cvtColor(). Next, sift.detect() and sift.compute() are used to extract the keypoints of the picture. After that, matcher.knnMatch() is used to match the keypoints. Finally, the matching results are filtered according to distance and an image is drawn according to the matching results.

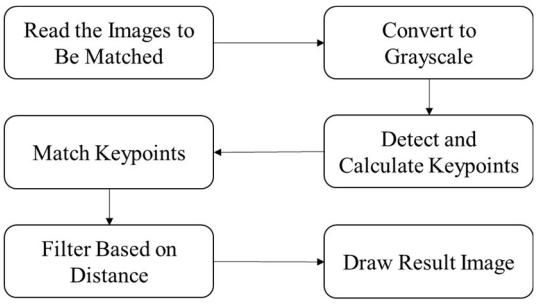


Fig. 1 SIFT Implementation Process

4.1.5. Android Development

This program sets up a common object database, including 7 types of common objects. It can match the objects of categories in the database. Layout sets a Button for each category of objects, which is used to load the object image of the corresponding category. With the Onclick event, an Intent object is created when the Button is clicked. It calls the file in the system album and specifies the RequestCode, which is used to specify the category of object recognition. The uri of the data returned by the Intent is obtained via data.getData() along with the requestCode. the image is passed to findImageSIFT() for matching. This function calls the findImageSIFT.js file, which is used to implement the SIFT algorithm.

4.2. Object Recognition App Based on YOLOX

4.2.1. Development Environment

The model training part of the project is implemented in Windows Environment. Windows version is windows 10. CUDA 11.3 and cudnn 8.4.0 is imported. Python is used for model training. Python version is Python 3.8. And the realization of deep learning is based on the Pytorch framework, Pytorch version 1.10.1. For Android app development, the development environment is Android Studio 2021.1.1.23.

4.2.2. Data Source

The dataset of this project is extracted from the MS COCO dataset. COCO can be used for image detection, semantic segmentation, image captioning and other tasks. It includes 200000 images and 80 target categories.
In this project, 38 types of common items in daily life are selected from the COCO dataset, including ties, mice, water cups, etc. The data extraction is completed based on the FiftyOne module. FiftyOne can be used to specify the qualified part of downloading large data sets such as COCO.

4.2.3. Model Training and Evaluation

Considering the computing power of mobile phones, the lightweight network models YOLOX-Tiny and YOLOX-Nano are selected for training.
First, YOLOX is cloned and the necessary dependencies should be installed according to requirements.txt. Then the pre-training weights can be downloaded from the official website. The dataset format must be configured according to the specified COCO dataset format. After that, the training configuration should be modified according to the actual demand. It is necessary to modify the depth and width of the network, the number of dataset categories and input image size. The number of epochs can also be modified in yolox_base.py. The default training epoch is 300 and it was not modified in the project.
According to **Fig. 2** and **Fig. 3**, the training results of YOLOX-Tiny and YOLOX-Nano are convergent. The AP of YOLOX-Tiny reached 32.26% in COCOAP50, and reached 32.26% in COCOAP5095. The YOLOX-Nano reached 36.75% and 21.93% respectively. Ideal training results were achieved.

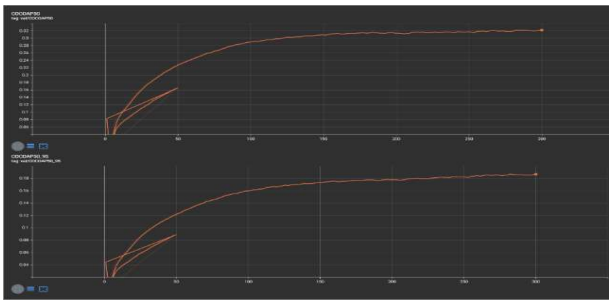


Fig. 2 Image of AP Changing with Epoch in YOLOX-Tiny under COCOAP50/ COCOAP50_95

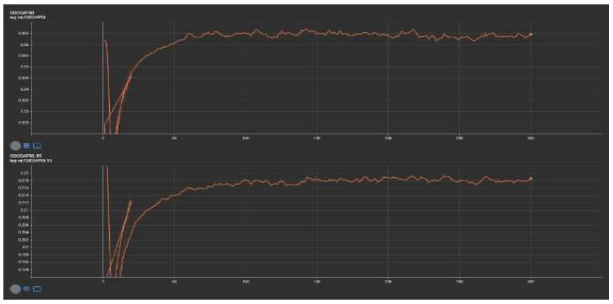


Fig. 3 Image of AP Changing with Epoch in YOLOX-Nano under COCOAP50/ COCOAP50_95

4.2.4. NCNN

NCNN is developed by Tencent. It is a high-performance neural network forward computing framework optimized for mobile terminals. It is an open-source neural network inference library [16]. Based on the NCNN, developers can perform cross platform operations to realize the migration of in-depth learning algorithms on the mobile end, and do not need to rely on a third party.

4.2.5. Algorithm Implementation

After the model training, the model file of YOLOX is obtained. In order to migrate it to the mobile terminal, we must first convert it into an open format ONNX model through the ONNX framework. Then, it is transformed into NCNN model file through NCNN framework, and optimized to obtain the recognizable file of mobile terminal. Finally, the NCNN file is imported into the Android project file and exported as APK. The implementation process is shown in **Fig. 4**.

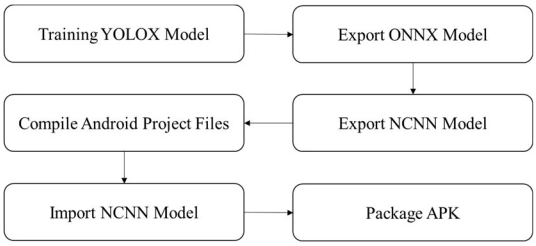


Fig. 4 YOLOX Implementation Process

4.2.6. Android Development

Layout uses linear layout. It switches the camera and selects the image via Button and switches the model and CPU/GPU via Spinner. And surfaceView is used to display the image. The program contains two Activities. MainActivity is used to switch the camera. An ncnnyolox object is generated via ncnnyolox(). View is drawn by onCreate() and openCamera() is called to open the camera. It is realised by changing the position of the current_model via the onItemSelected() function when the model is toggled by clicking on the Button. If the Button to select an image is clicked, it will jump to the ImageSelectActivity using Intent. The structure of both Activities is basically the same.

5. PROGRAM TESTING

5.1. Object Recognition App Based on SIFT

The SIFT object image matching program UI is shown in **Fig. 5**. After clicking the object category to be recognized, the system will jump to the system photo album and wait for the image to be matched to be extracted.



Fig. 5 App Interface Based on SIFT

After selecting an image, the program will automatically start matching. If the matching is successful, the matching result image will be automatically opened through the system album. Otherwise, it will return to the initial interface of the program. The matching results of some images are shown in **Fig. 6**.

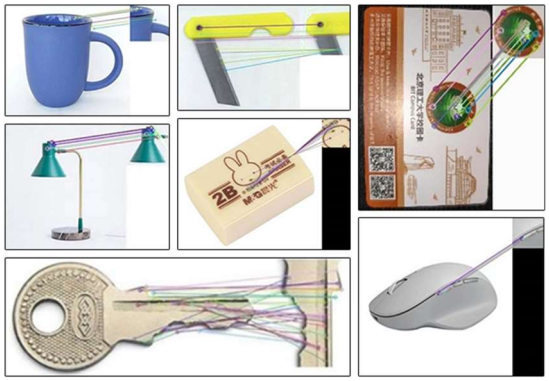


Fig. 6 SIFT Image Matching Results

It can be seen from the image matching results that the app can correctly realize the image feature matching. Moreover, it can establish accurate image feature matching when dealing with image rotation, scaling and rotation scaling.

However, when dealing with complex scenes, SIFT is difficult to establish accurate matching of image features. Moreover, when the SIFT algorithm is run on the mobile phone, the operation speed will be very slow if the size of the input picture is large. And there may be a memory leak, causing the program to flash back.

5.2. Object Recognition App Based on YOLOX

The program UI and running results are shown in the figures. The front and rear cameras are switched by clicking the "Switch Camera" button. The image in the album for recognition is selected by clicking the "Select Image" button. The model and CPU/GPU can also be switched by clicking the corresponding spinner.

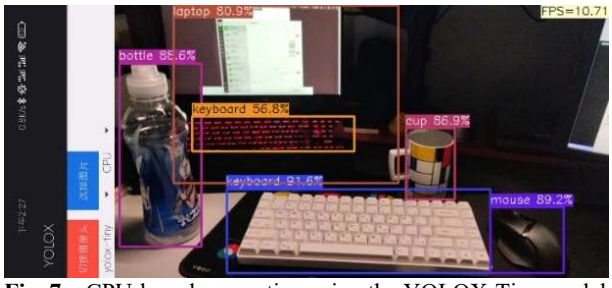


Fig. 7 CPU-based computing using the YOLOX-Tiny model



Fig. 8 GPU-based computing using the YOLOX-Tiny model

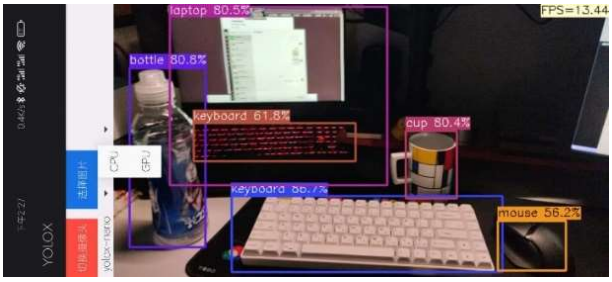


Fig. 9 CPU-based computing using the YOLOX-Nano model

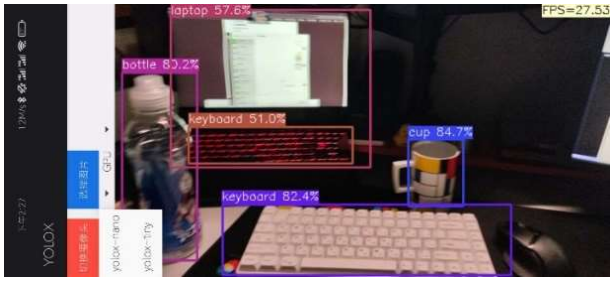


Fig. 10 GPU-based computing using the YOLOX-Nano model

The results show that both models have good detection accuracy. When both models use CPU or GPU, the lighter network model YOLOX-Nano has faster operation speed than YOLOX-Tiny according to the frame rate of target detection (in **Fig. 7** and **Fig. 9**), because of their different numbers of parameters. When the same model is used for prediction, the frame rate shows that using the GPU for computing is faster(in **Fig. 7** and **Fig. 8**).

To test the detection accuracy of the app, a 14 second video and 160 pictures of common objects in life were collected. Manually confirm the recognition result of the app. In all 439 frames of video images, the app correctly recognized 422 frames, with a recognition accuracy of 96.1%. In all 160 images, the app correctly recognized 148 images, with a recognition accuracy of 92.5%. After verification, the recognition accuracy of app has exceeded 90%.

6. CONCLUSION

Based on the phenomenon of looking for personal items in daily life, this paper designs two Android apps based on SIFT and YOLOX to realize the function of object search.

The innovation of this paper is: Based on the NCNN framework, this paper implements the encapsulation of the training model. It makes it possible to predict directly at the mobile and realize real-time object recognition. Its advantage is that it gets rid of the dependence of the network in the online mode. And it removes the restriction of network speed on the real-time performance of object recognition. At the same time, it can protect the privacy of users.

The limitation of this article is that both apps can only recognize some categories of objects. They do not have the ability to identify objects that do not collect data. Therefore, they have certain limitations in the use process.

REFERENCES:

[1] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]//2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). Ieee, 2005, 1: 886-893.

[2] Ojala T, Pietikainen M, Maenpaa T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns[J]. IEEE Transactions on pattern analysis and machine intelligence, 2002, 24(7): 971-987.

[3] Li S, Wei F. Logo Recognition System on Mobile Terminal[J]. Computer engineering & Software, 2016 (2): 98-102.

[4] Shao Z, Yao Q, Tang J, et al. Research and Development of the Intelligent Identification System of Agricultural Pests for Mobile Terminals[J]. Scientia Agricultura Sinica, 2020,53(16):3257-3268.

[5] Lowe D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60(2): 91-110.

[6] Iijima T. Basic theory on the normalization of pattern (in case of typical one-dimensional pattern)[J]. Bulletin of Electro-technical Laboratory, 1962, 26: 368-388.

[7] Al-Shuibi A, Aldarawani A, Al-Homaidi H, et al. Survey on Image Retrieval Based on Rotation, Translation and Scaling Invariant Features[C]//2019 First International Conference of Intelligent Computing and Engineering (ICOICE). IEEE, 2019: 1-11.

[8] Babaud J, Witkin A P, Baudin M, et al. Uniqueness of the Gaussian kernel for scale-space filtering[J]. IEEE transactions on pattern analysis and machine intelligence, 1986 (1): 26-33.

[9] Lindeberg T. Scale-space theory: A basic tool for analyzing structures at different scales[J]. Journal of applied statistics, 1994, 21(1-2): 225-270.

[10] Wang S, Liu Y, Li D, et al. An improved SIFT feature extraction method for tyre tread patterns retrieval[C]//2014 Seventh International Symposium on Computational Intelligence and Design. IEEE, 2014, 1: 539-543.

[11] Li Y, Shi G. ORB-based Fingerprint Matching Algorithm for Mobile Devices[C]//2019 IEEE 2nd International Conference on Computer and Communication Engineering Technology (CCET). IEEE, 2019: 11-15.

[12] Liu G, Liu S, Liang L, et al. Scene Matching for Infrared and Visible Images with Compressive Sensing SIFT Feature Representation in Bandelet Domain[J]. International Journal of Pattern Recognition and Artificial Intelligence, 2020, 34(11): 2054029.

[13] Ma S, Lu H, Wang Y, et al. YOLOX-Mobile: A Target Detection Algorithm More Suitable for Mobile Devices[C]//Journal of Physics: Conference Series. IOP Publishing, 2022, 2203(1): 012030.

[14] Liu X, Zhou H. A Chinese Small Vocabulary Offline Speech Recognition System Based on Pocketsphinx in Android Platform[C]//Applied Mechanics and Materials. Trans Tech Publications Ltd, 2014, 623: 267-273.

[15] Jiang L, Jiang X, Chen X, et al. Application of OpenCV in License Plate Recognition[J]. Computer engineering & Software, 2014(3):45-47.

[16] Yang D, Yang L, Liu Y. Research and Implementation of Embedded Real-time Target Detection Algorithm Based on Deep Learning[C]//Journal of Physics: Conference Series. IOP Publishing, 2022, 2216(1): 012103.