

Design and Implementation of a Real-time Target Detection and Tracking System

Xinyan Cao*, Guang Chen**, Yaxin Zhao**, Wei Ren**, Jian Cao*

*Department of Computing, The Hong Kong Polytechnic University, Hong Kong of China

**School of Software and Microelectronics, Peking University, China

21049228g@connect.polyu.hk, chguang@pku.edu.cn, yxzha@stu.pku.edu.cn, renw@stu.pku.edu.cn, caojian@ss.pku.edu.cn

Abstract— Target detection is to quickly and accurately locate the object to be measured in the video image and correctly classify it. Target tracking is to obtain the motion parameters of the object in the image sequence, and to detect, extract, identify and track the specific object. Combining target detection and tracking to follow specific objects is one of the research hotspots in the realization of intelligent embedded terminals. This paper designs and implements a system that combines target detection and tracking, which can realize real-time tracking of specific objects. Finally, the experiments prove the reliability of the proposed system, which can be used in intelligent systems such as robots.

Keywords— Target detection, Target detection, Deep learning, Real-time System, YOLOv3-tiny

I. INTRODUCTION

In recent years, target detection models and target tracking algorithms have become a hot topic of interest as key components of deep learning. The target detection task locates the object to be measured in the video image quickly and precisely and classifies it correctly. Target tracking is the detection, feature extraction, recognition and tracking of a specific object with the aim of obtaining the parameters of the object's motion in the image sequence. Combining target detection and tracking to track a specific object is the key to achieve an intelligent system.

In this paper, YOLOv3-tiny in [1] and [2] is chosen as the target detection model and KCF is chosen as the target tracking algorithm. Based on the homemade PKUAIP dataset, this paper deploys the YOLOv3-tiny target detection model on the embedded platform NX to detect the object of the video input from the camera, find the position and size of the red ball, and then supplement it with the target tracking algorithm KCF to track the position of the red ball in real time, and NX then communicates with the Mecanum wheel car through the serial port to control the vehicle for left/ right/ forward/ backward motion. Finally, the real-time following motion of the red ball is realized.

The structure of the paper is as follows: the first part is the preface. The second part is the target tracking algorithm study. The third part is the system design scheme, the specific design and implementation. The fourth part is the experiment and result analysis. The fifth part is the conclusion.

II. TARGET TRACKING ALGORITHM

A. Traditional Target tracking Algorithm

Reference [3] shows Henriques and others then proposed a KCF target tracking algorithm based on kernel ridge regression that mentioned in [4], which is known for its accuracy and high speed, and uses histograms of oriented gradients (HOG) features for feature extraction, and selects one of the three kernels of Gaussian, linear and polynomial kernels. The accuracy and speed of these three different kernels are different, compared with linear kernels, Gaussian kernels have the highest accuracy and slower speed. During the tracking process, the tracker can find the object accurately without occluding it, but how to cope with the fast motion is still a challenge for the present tracking algorithm.

The KCF algorithm uses ridge regression to train the target detector, collects positive and negative samples using a circular matrix of the region around the target, and transforms the operations of the matrix into dot product of the elements, substantially increasing the speed of the operations and allowing the algorithm to meet real-time requirements. Another contribution of KCF is the groundbreaking incorporation of multiple channels of data compared to traditional target tracking algorithms. Compared to other target tracking algorithms, the KCF algorithm is much faster and more effective than other target tracking algorithms, and the algorithm is relatively simple to think about and implement.

In reference [5] and [6], Kalman filters were proposed by Rudolph Kalman and others in the 1960s and are mainly used to estimate the parameters (or states) of linear Gaussian dynamical systems. Kalman filters are Bayesian optimal solutions that can be used to sequentially estimate the states of dynamical systems. In this system, the state evolution and measurement process are both linear and Gaussian, while the Kalman filter can be used to estimate the position and velocity of a moving object.

B. Deep Neural Network-based Target tracking Algorithm

Recently, deep convolutional neural networks (CNNs) have shown impressive performance in many tasks, and several studies have successfully applied convolutional depth features to a related filter framework, but it is not possible to combine multi-resolution functions on this framework. To address

these problems, continuous convolution operator filter (C-COT) in [7] proposes an implicit interpolation model using training samples, which solves the object localization problem at different resolutions, while C-COT also penalizes the filter using spatial regularization, so that the penalty coefficient of the filter corresponding to the object area is small, and the background coefficient is large, so as to solve the boundary effect. Recently, the DCF discriminative correlation filter approach using intra-network feature hierarchy in convolutional neural networks has achieved recent results in visual tracking, but their models are complex and cannot run in real time.

C. Our Target tracking System

Target tracking algorithm based on deep learning is one of the important theoretical supports for car tracking red ball experiments. The application scenario of the experiments in this paper is target detection for red balls, which is not demanding in terms of accuracy. On the contrary, the speed of the model is required in order to track red balls quickly in the subsequent steps. Therefore, under the premise of comprehensive consideration of various algorithms, this paper selects the mature algorithm YOLOv3-Tiny model of YOLO series, and combine with the KCF model to implement our system.

III. SYSTEM DESIGN

The system is run in NX development board, and the red ball detection and tracking system is deployed in the embedded terminal. The whole system includes image acquisition module, NX development board, red ball target detection module, red ball target tracking module, car control module, data communication module, image display module and car motion module. The complete red ball detection and tracking system is built from the above 8 modules.

A. System Design

The system design is divided into two parts: software design and hardware design for detailed description.

The overall design of the red ball detection and tracking system is shown in Figure 1. The red ball target detection model is trained based on the server. As shown in the figure 1, the NX development board is the core of the whole embedded terminal deployment. The image information is collected by connecting with the camera, and the detection model is ported to the NX development board after accelerating the YOLOv3-tiny model by using TensorRT, which outputs the red ball positioning and size information to the LCD after target detection. NX relies on the UART serial port to communicate with the motion module of the car to control the left/ right/ back/ forward movement of the car. The following key components constitute the entire system:

1) **NX Development Board:** NX is the core part of the entire embedded system.

2) **Image Acquisition Module:** This part transmits the acquired image information to the NX platform through the camera.

3) **Image Display Module:** The display is connected to the NX via the micro-HDMI interface, and is responsible for displaying the collected images and the red ball detection and tracking results.

4) **Red Ball Detection Module:** Target detection based on the YOLOv3-tiny model is used to locate the position and size of the red ball on the camera input video.

5) **Red Ball Tracking Module:** This module mainly implements the function of tracking of the red ball position in real time.

6) **Data Communication Module:** NX communicates with Mecanum wheel car via UART serial transmission.

7) **Car Control Module:** This module controls the car for left/ right/ back/ forward motion.

8) **Car Motion Module:** This part realizes the movement of the car following the red ball.

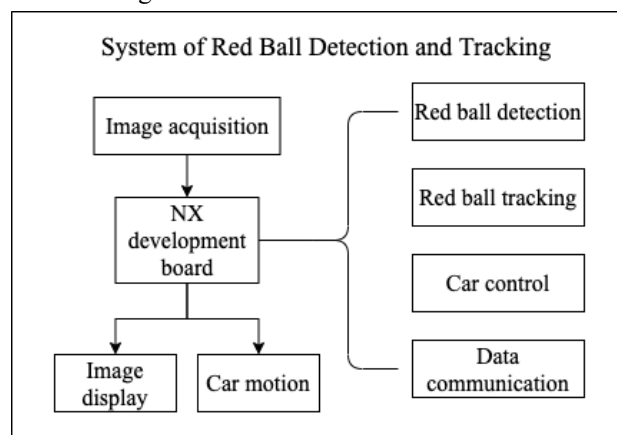


Figure 1. Overall design block diagram of red ball detection and tracking system

B. System Specific Design and Implementation

The system design is divided into hardware part and software part. The hardware part of the system is designed with a camera、Mecanum wheel car、NX development board、LCD monitor and other devices. The whole system design is based on the embedded deployment platform, through the camera to capture images after the input image preprocessing and transfer to the NX platform, and then the camera input video for target detection, while locating the position and size of the red ball supplemented by target tracking algorithm to track the position of the red ball in real time, finally, the results will be presented on the LCD display. NX through the UART serial port and Mecanum wheel car for communication. The car is controlled to move left/ right/ back/ forward to finally achieve the following motion of the red ball.

The software part of this system contains a software environment using TensorRT to optimize the model for acceleration, an OpenCV library is selected to preprocess the image input from the camera, and the PyTorch framework is chosen for feature extraction.

In this paper, we adopt the frame per second (FPS) of the image processed by the model algorithm as an indicator to measure the performance of the red ball target detection model. The red ball target detection and tracking system is designed with the fundamental goal of achieving the highest possible model representation ability under limited computational complexity, and to increase the actual execution speed of the model. After the preliminary dataset preparation and hardware design, the software part is designed and the overall software flow of the red ball detection and tracking system is shown in Figure 2.

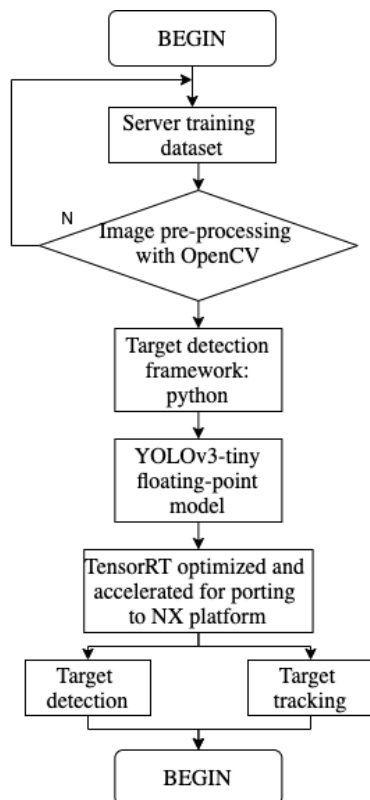


Figure 2. Red ball detection and tracking system software design block diagram

After training the self-made PKUAIP sphere dataset using the YOLOv3-tiny target detection model in the server, preprocess the camera input images by using the OpenCV library, and repeat the training and image preprocessing of the dataset. If this step fails, repeat the training and image preprocessing of the dataset, and then transport the preprocessed images to the target detection framework PyTorch for feature extraction, and the extracted features are fed into the YOLOv3-tiny model; TensorRT is used to optimize and accelerate the target detection model, and the processed model is ported to the embedded deployment platform NX; the images input to the camera are preprocessed and sent to NX, and the video input to the camera detects and locates the position and size of the red ball in a object manner, and is supplemented by the KCF target tracking algorithm to track the position of the red ball in real time. So far, the software design of the red ball detection and tracking system is completed.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. The Dataset

The experiment uses a self-made PKUAIP sphere dataset and uses the YOLOv3-tiny target detection model for training the dataset. The dataset is divided into a training set and a validation set, each containing 1535 instances that have been labeled. It contains not only the dataset and labels in yolo format, but also the data training parameter files and cfg network parameter files required for training the PyTorch framework. The image annotation consists of a bounding rectangle with red balls that are clearly distinguishable to the human eye in each of the images present in the dataset. No restrictions on the location of the balls and the scene brightness were taken in the production of the dataset.

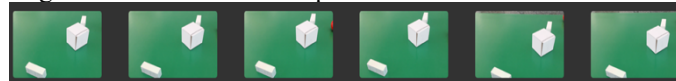


Figure 3. Example diagram of dataset

B. Experimental Environment Construction

The experiment is based on PyTorch, which is a deep learning framework that supports multi-card parallel training to build the YOLOv3-tiny target detection model, and uses the CUDA library to accelerate the graphics card computation. The experimental host system is Ubuntu 18.04.5 LTS, the host processor is Intel Xeon Gold 6149, and NVIDIA RTX 2080Ti graphics card is used to meet the model training requirements in this experiment. In addition, NX requires the installation of common libraries such as OpenCV and Jet Pack. In order to run the target detection model smoothly on the NX platform, the trained model needs to be accelerated by TensorRT.

C. Terminal Platform Deployment

The fundamentally purpose of the experiments in this paper is to deploy the YOLOv3-tiny model with KCF algorithm to the embedded terminal NX. The data flow architecture deployed by the model terminal is shown in Figure 4.

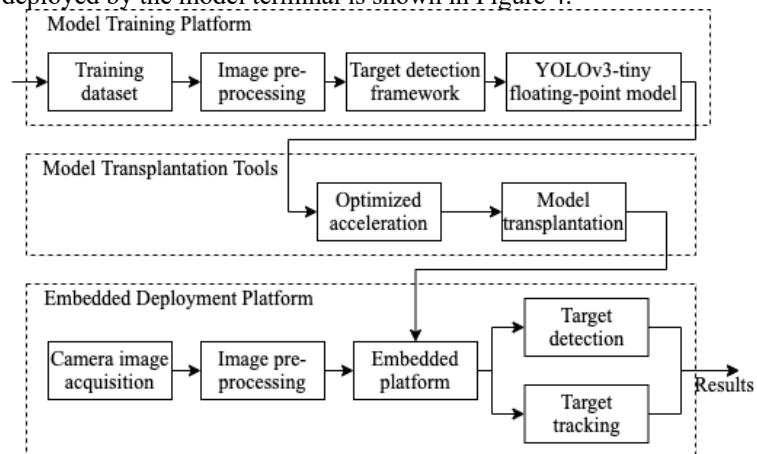


Figure 4. Model terminal deployment data flow diagram

The hardware and software environment and the preparation of relevant data sets are required in the first stage. The data flow architecture of the model terminal deployment consists of three modules.

The first module: server-based to complete this part of the experiment. First train the dataset, preprocess the images and input them into the target detection framework to extract features, and then feeding the extracted features into the YOLOv3-tiny floating-point model.

The second module: based on the model migration tool. Using TensorRT to optimize the acceleration of the YOLOv3-tiny floating-point model, and transplant the processed model to the embedded deployment platform.

The third module: based on the embedded deployment platform. The images are captured by the camera, and the input images are preprocessed and transmitted to the NX platform. The location and size of the red ball are found after performing the target detection on the video input by the camera, and then supplemented with target tracking algorithm to track the position of the red ball in real time. NX then communicates with Mecanum wheel car through the UART serial port to control the left/ right/ back/ forward movement of the car.

D. Analysis of Experimental Results

The measured results of the target detection model based on the NX platform are shown in Table 1, and the target detection results are shown in Figure 5.

TABLE 1. YOLOV3-TINY MODEL TRAINING REAL-WORLD RESULTS (RTX 2080Ti×4)

YOLOv3-tiny Floating-point Model		
Reasoning time	FPS	mAP
8.69ms	115.07	0.995

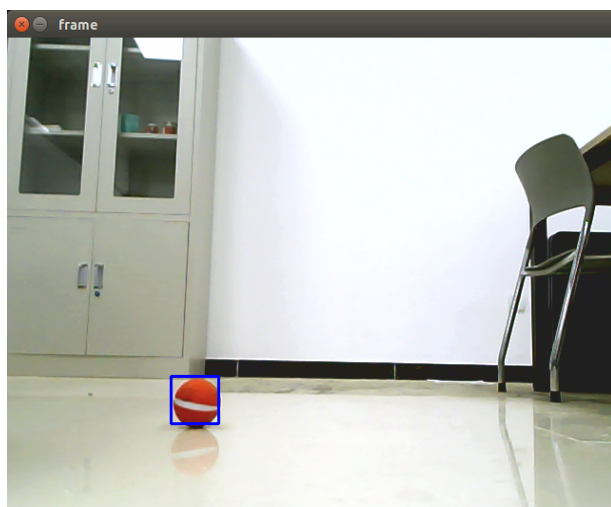


Figure 5. Target detection results

The measured results of the target tracking algorithm based on the NX platform are shown in Table 2, and the results of NX operation are shown in Figure 6.

TABLE 2. MEASURED RESULTS of the TARGET TRACKING ALGORITHM (XAVIER NX)

KFC Target tracking Algorithm	
Reasoning time	FPS
19.95ms	50.12

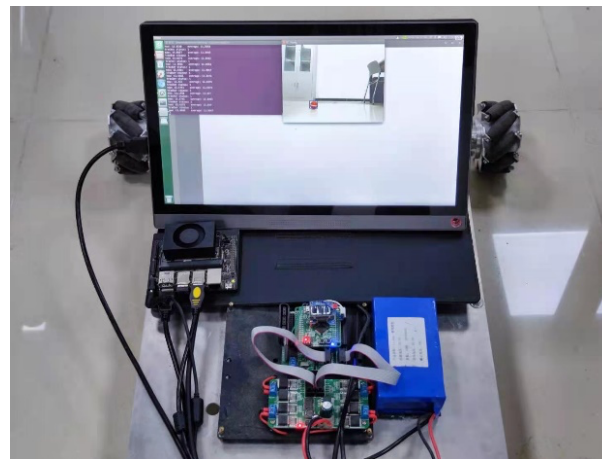


Figure 6. NX run results

The test results of terminal deployment based on NX platform YOLOv3-tiny model are shown in Table 3, and the terminal deployment results of the entire system are shown in Table 4.

TABLE 3. ACTUAL RESULTS of YOLOV3-TINY TERMINAL DEPLOYMENT (RTX 2080Ti×4)

Algorithm Type	Reasoning Time	FPS
YOLOv3-tiny floating-point model TensorRT acceleration model	16.46ms	60.75
YOLOv3-tiny semi-floating point TensorRT acceleration model	11.12ms	89.93

TABLE 4. YOLOV3-TINY TARGET DETECTION and KCF TARGET TRACKING SYSTEM TERMINAL DEPLOYMENT REAL-WORLD TEST RESULTS (XAVIER NX)

YOLOv3-tiny Target detection and KCF Target tracking System	
Reasoning time	FPS
19.95ms	61.35

The experimental results show that the target detection model for red ball reached 89.93 FPS; the target tracking model for car tracking red ball reached 50.12 FPS; let the YOLOv3-tiny target detection and KCF target tracking system run for one minute and take its average speed, then as shown in Table 4, it shows that the total inference speed of the system on the NX platform has reached 61.35 FPS.

V. CONCLUSIONS

In this paper, a real-time target detection and tracking system has been possible. However, there is still much room for research on this system. The following are some directions worth exploring.

Firstly, in real scenes, the detection accuracy of the target detection model is affected to varying degrees due to disturbances in the surrounding environment, such as changes in light. The overall performance of the model can be improved by, for example, adding defogging pre-processing. Secondly, the shape of the object to be detected varies greatly,

and this experiment uses a home-made PKUAIP dataset with a red ball as the object to be detected. Multiple datasets can be used for comparative validation and comparison with existing techniques to highlight the practicality of the model proposed in this paper. Thirdly, an infrared detection device can be added to the system to enable the monitoring and tracking of live animals, which can be applied to the monitoring of animal activities in the field. Finally, for the target detection model and tracking algorithm in this paper, further compression can be applied by means of model pruning and quantization. With the same model accuracy, the number of floating point operations is reduced and the overall efficiency of the system is improved.

ACKNOWLEDGMENT

I am very grateful to my fellow students and teachers for their sincere help and encouragement, without whom I would not have been able to complete this thesis. I am very grateful to my thesis supervisor, Associate Professor Cao Jian, who guided the direction of this paper and helped me to continuously revise and improve it. In addition, I would like to thank my classmates Feng Lin, Guang Chen, Yaxin Zhao and Wei Ren for their contributions, which brought together all of us for our hard work to make this paper possible.

VI. REFERENCES

- [1] Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger. The IEEE Conference on Computer Vision and Pattern Recognition. Hawaii: IEEE Computer Society, 2017: 6517-6525.
- [2] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement. <https://arxiv.org/abs/1804.02767>, 2018-04-08.
- [3] B. Babenko, M.-H. Yang and S. Belongie, "Visual tracking with online Multiple Ins.
- [4] J. F. Henriques, C. Rui, P. Martins and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 3, pp. 583-596, 2014.
- [5] R. Kalman, "A new approach to linear filtering and prediction problems" Trans. ASME Ser. D. J. Basic Eng., vol. 82, pp. 35-45, 1960. DOI: 10.1115/1.3662552.
- [6] R. Kalman and R. Bucy, "New results in linear filtering and prediction theory" Trans. ASME Ser. D. J. Basic Eng., vol. 83, pp. 95-108, 1961. DOI: 10.1115/1.3658902.
- [7] M Danelljan, A Robinson, F S Khan et al., "Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking", European Conference on Computer Vision, 2016.



First A. Xinyan Cao, is studying for her M.S. degree at the Hong Kong Polytechnic University, majoring in the Department of Information Technology, and her research direction is in the field of computer vision and big data analysis. In the past few years, she has published several articles in the field of computing. Cao received her first B.Eng. degree in Communications Engineering from Beijing Union University, and completed a second bachelor's degree in Software Engineering from Peking University, supervised by Profs. Jian Cao.



Second A. Guang Chen, a Ph.D. candidate in the School of Software and Microelectronics of Peking University, majoring in electronic information, and researching in target detection and artificial intelligence system. He received a B.Eng. degree in Electronic Information Engineering from Xidian University and a M.Eng. degree in Electronics and Communication Engineering from Peking University.



Second B. Yaxin Zhao, a postgraduate student in the School of Software and Microelectronics of Peking University, majoring in Electronic Information, and researching in computer vision and embedded systems. She received B.S. degree in Computer Science and Technology from the Computer School of Wuhan University. And won the highest scholarship of Wuhan University in 2019, Yu Gang Song Xiao Scholarship. In 2020, she joined the SOC Digital Multimedia Lab at Peking University.



Second C. Wei Ren, received a master's degree in Software Engineering from Peking University in 2013. He is currently studying for a Ph.D. degree at Peking University and is also the president of C4 Games. Mr. Ren's current research interests focus on construction and application of brain-computer interfaces.



Third A. Jian Cao, an associate professor in Peking University. He has devoted himself to the research of artificial intelligence and robotics. The research fields include: neural network model compression and deployment, artificial intelligence algorithm optimization and implementation, computer vision, edge computing. He has published more than 50 journals and conference papers indexed by SCI and EI, and applied for more than twenty patents. He received a master degree in Software Engineering, and a Ph.D. degree in Microelectronics and Solid-State Electronics from Peking University.