Deep Learning for Object Identification in ROS-based Mobile Robots

Yeong-Hwa Chang, Ping-Lun Chung, and Hung-Wei Lin
Intelligent Control Laboratory
Chang Gung University
Taoyuan, Taiwan

Abstract

In this paper, an open source robotic middleware ROS is applied to identify objects with a Raspberry Pi based mobile robot. Faster R-CNN algorithm is considered for enhanced deep learning. The capabilities of robot control and object detection are verified through experimental validations. Based on a low-cost Raspberry Pi control kernel, an easy-to-use interface is developed to integrate front-end and back-end applications. Firstly some ROS packages for the mobile robot are needed to be designed. Then the captured environment information will be provided for objective detection using GPU-accelerated computing. In this paper, Kinect sensor is used to capture images and the deep learning algorithm is implemented based on ROS middleware. All previous mentioned function modules are contained in a cloud service.

Keywords: ROS; Deep Learning; Object Detection; Robot Control; System Integration

I. Introduction

Traditional service robots [1, 2] are usually lack of cognitive services such as image and voice identification. There still exists a clear gap how to integrate service robots with human intelligence. Recently, machine learning techniques have been widely discussed. There are many remarkable progresses that can be used to fulfill cognitive services. Deep learning algorithms have been popularly addressed that may lead to a new generation in machine learning discipline. For example, a deep learning neural network was applied for hand written recognition [3]. In recently years, deep learning methods have emerged as powerful machine learning methods for object recognition and detection. Compared with traditional approached, deep learning methods can quickly learn the features directly from the raw images. In deep learning algorithms, Faster R-CNN (Region Convolution Neural Network) is one newly developed algorithm [4]. According to different datasets, Faster R-CNN was implemented for object detection [5].

Object recognition and detection is a key problem in the area of cognitive service, especially in moving objects [6]. Traditional object recognition methods basically focused on improving recognition performance on specific datasets. In general, PASCAL VOC, Microsoft COCO, and ImageNet are three popularly mentioned datasets [7-9]. These datasets provide sufficient variability in object categories and instances, the training data mostly consists of images of arbitrarily picked scenes and objects. If mobile robots can integrate with deep learning based object detection, more interesting applications could be generated. Robot Operating System (ROS) is a robotic middleware with many open resource packages [10]. It will be more efficient to develop applications using a ROS framework. In [11], the control of ROS-based mobile robot was addressed using Arduino platform. In past few years, mobile robot integrated with deep learning method based on ROS middleware has attracted a lot of attention, such as the applications on shopping, navigation, and environment recognition [12-14].

In this paper, object detection and recognition can also be combined with mobile robot which is setting on ROS middleware. Some ROS packages are designed for the mobile robot to attain the desired tasks, such as robot control and simultaneously localizing and mapping. Based on Raspberry Pi, an easy-to-use control kernel is developed to integrate front-end and back-end applications. In front-end part, the raw images captured by the Kinect sensor are provided for object detection by using Faster R-CNN algorithm and GPU accelerated computing. The improved learning model can be obtained by deep learning framework. Deep learning methods closely rely on certain big data. In this paper, Microsoft COCO and PASCAL datasets are utilized for Faster R-CNN learning. Different cases will be investigated to verify the feasibility of the proposed scheme. From experimental validations, the recognition is quite satisfied.

In this paper, the Faster R-CNN learning algorithm is briefly explained in Sec. 2. The experiment setting is described in Sec. 3. The experimental results and integration testing are discussed in Sec. 4 and Sec. 5, respectively. Concluding remarks are given in Sec. 6

II. Preliminaries of Deep Learning Algorithms

A. R-CNN and Fast R-CNN [15, 16]

R-CNN is a visual object detection algorithm, combining region proposals and convolutional neural networks. At the time of its release, R-CNN improved the previous best detection performance on PASCAL VOC 2012 by 30%, going from 40.9% to 53.3% mean average precision. Unlike the previous best results, R-CNN achieves this performance without using contextual rescoring or an ensemble of feature types. On the other hand, Fast R-CNN is a fast framework for object detection with deep convolution network. Fast R-CNN proposed a clean and fast update to R-CNN. This is the first time to use GPU, but the part of region-based feature extraction is on CPU.

B. Faster R-CNN [4, 5]

Faster R-CNN is efficient for accurate region proposal generation. Faster R-CNN can be easily defined as a system with region proposal network (RPN) and Fast R-CNN. This model has significant improvement compared to R-CNN and Fast R-CNN approaches. The RPN, sharing the convolutional layers with the detection network, leads to real-time performance at inference time and allows an efficient training procedure. This paper compared strength and weakness with

ISBN 978-1-5386-4342-6

three types of R-CNN so in this article implement Faster R-CNN to integration our system and have a proof of concept for our scenario. We survey and compare three papers and select good one to implement and integrate with system. We chose the region-based method Faster R-CNN as a representative architecture among the ones recently proposed in the Deep learning literature for the same task.

C. Dataset

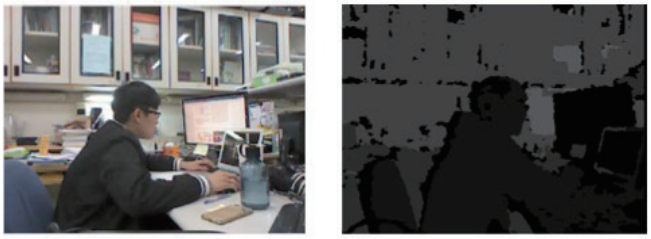
In general, PASCAL VOC, Microsoft COCO, and ImageNet are three popular datasets, often used for neural network training. The Pascal VOC challenge is a very popular dataset for building and evaluating algorithms for image classification, object detection, and segmentation. COCO is a large-scale object detection, segmentation, and captioning dataset. The advantages about Microsoft COCO dataset provided for object segmentation, Recognition in context, multiple objects per image, more than 300000 images, more than 2 million instances and so on. ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently It have an average of over five hundred images per node. Hoping ImageNet will become a useful resource for researchers, educators, students and all of you who share our passion for pictures.

III. Proposed System

In this paper, Raspberry Pi is used as the control kernel for mobile robot. The system runs on the Ubuntu MATE operating system using ROS middleware. The proposed system was implemented using Python programming language so that the Kinect sensor can capture environment information and upload to a SQL database. Then the captured images will be provided for objective identification. In addition, a chatbot will be designed to real-time reply the inquiries from users regarding the one particular detected object. To execute the whole data processing, some databases will be created for sequent works. Firstly, raw images are captured from Kinect sensor, shown as Fig. 1. Then the tested images will be provided for object identification. It is usually time-consuming in the deep learning recognition process. In this paper, a more effective learning algorithm Faster R-CNN will be used to identify objects such that the real-time application becomes feasible. The details about the proposed system functions will be discussed in the following.

A. Object Detection and System Integration

The proposed system architecture is shown in Fig. 2, including the ROS mobile robot and object detection. The ROS operation system on Raspberry Pi supports the Kinect driver to capture images and connect to cloud service via Wi-Fi. In the part of object detection, GPU is used to accelerate the processing of Faster R-CNN learning algorithm. The mobile robot first capture images from Kinect sensor, and the image data will be real-time analyzed by GPU processing. Then the classified objects will be stored in SQL blobs. Through the recognition process, the detection probability of a specific object can be also obtained. To fulfill more complicated tasks, some deep learning framework is adopted, such as Caffe [17]. Caffe is a deep learning framework that can be used to construct neural networks. In practice, Caffe supports cuDNN v5 for GPU acceleration that is the main benefit by using this framework. According to Fig. 2, the captured images will be stored in a raw image folder. The Faster R-CNN module will analyze these updated images, which means the analyzing routine will be continuously performed as long as new raw data adding on. Based on recorded time instants, the recognition results will be uploaded to corresponding object result folder. In practice, the object detection results will be different with respect to target objects and chosen datasets.



(a) (b)

Fig. 1 Real-time image capture: (a) RGB image, (b) depth image.

B. ROS Node Processing

The essential idea of the ROS-based image processing is shown in Fig. 3, where each node represents one task and all the nodes can be executed in distributed mode. To complete the node processing, one can develop necessary scripts according designed tasks. For example, the mobile robot may be desired to move around and capture environmental images simultaneously. In this paper, one particular script is developed to capture images and put a time stamp on each image. Also, these tagged images will be send to GPU for object detection. Before running this script, the following steps used to communicate Raspberry Pi with ROS need to be executed: (1) start the ROS system, and (2) execute launch file. The command lines for these two steps are roscore and roslaunch freenect_launch freenect.launch. According to previous explanation, the whole system will be more stable due to the ROS running on multiple machines. Unlike the scenarios one script containing only one packaged procedure, the ROS-based processing will be more to debug.

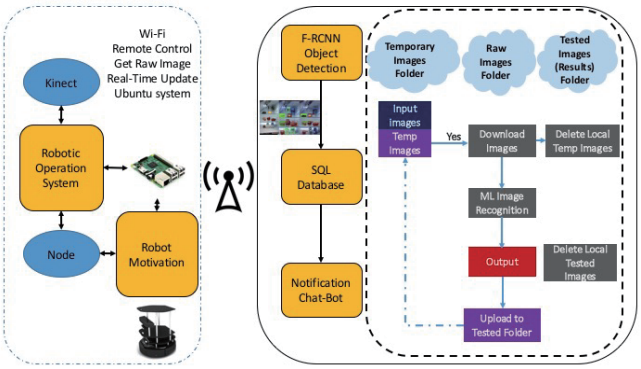


Fig. 2 The whole scheme of the proposed object detection system with ROS mobile robot.

ISBN 978-1-5386-4342-6

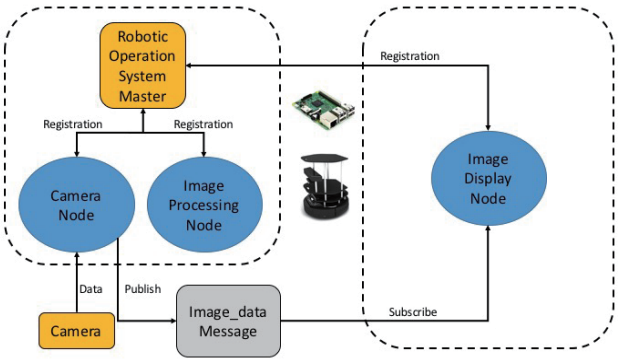


Fig. 3 The scheme diagram of ROS-based image processing.

IV. Implement on Faster R-CNN

In this paper, the object detection system is based on a Faster R-CNN learning network. The Faster R-CNN based object detection is basically composed of the following three essential steps, region proposal, feature extraction, and classification. These three steps will be finally integrated into a deep learning neural network. The Faster R-CNN algorithm can be considered as the combination of region proposal networks and Fast R-CNN system. In this paper, the proposed Faster R-CNN scheme is used for object detection with the integration with a ROS-based mobile robot. Firstly, the Kinect sensor captured images are viewed as the input images filtered in convolutional layers. Then, the region proposal network will provide feature maps. Finally, the classification step predicts object items. In this paper, the Simonyan and Zisserman network model (VGG-16) [18,19], having 13 shareable convolutional layers, is used as the testbed to fulfill the Faster R-CNN learning algorithm.

The details about how to implement Faster R-CNN will be discussed in the following:

A. Python Scripts and GPU Acceleration

Python language is adopted to write the script that includes SQL database and Faster R-CNN algorithm. From the raw images captured from Kinect sensor, the script is going to load all the raw image files for object analyzing. For real-time application, computing time comsumption is a key issue to be considered. To accelaerate the tedious processing runtime, Compute Unified Device Architecture (CUDA) is adopted. The CUDA [20] architecture is a parallel computing platform and an application programming interface (API) model created by NVIDIA. It allows software developers and software engineers to use a CUDA-enable GPU for general purpose processing. The script will be launched by using cloud service or secure shell (SSH) connection to control GPU system.

B. Learning Network

The ImageNet dataset is used to for pre-training the VGG-16 model as the base network. The consequent analysis basically follows the default setups: (1) Base network: The feature map from conv5_3 are used for region proposals and fed into region-of-interest (ROI) pooling. (2) Datasets: Both PASCAL VOC and COCO are used for evaluation. (3) Training/Testing: Following COCO challenge requirements, for each testing image, the detection pipeline provides at most 100 detection results. (4) Evaluation: The deep learning toolkit Caffe is evaluated by provided datasets. The evaluation metrics are basically based on the detection average precision.

C. Region Proposal Network (RPN)

RPN is a fully-convolutional network used to predict the possible region containing a certain object. The detected region will be sent to CNN to identify which the object is. With the RPN pre-processing, the object detection processing time can be significantly reduced. Due to the fully- convolutional behaviors, the RPN network has the advantage that different layers can share image features. Thus the object detection precision can be further improved. In this paper, the RPN is built by using the open-source packages in Caffe toolkits. The RPN network is then used to pre-filter the images captured by Kinect sensor.

D. Intersection over Union (IoU)

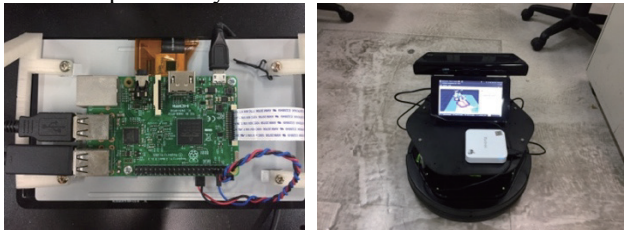
The region detected by RPN can be further fine-tuned by IoU. According to some designated recall rate, the boundary of the detected region can be adjusted. The precision of the desired object recognition can be increased. In this paper, the region boundary parameters need to be tuned with respect to recognition results. In general, the tuning process is related to the chosen dataset, task objects, and raw images.

V. Experiment and Results

The experimental setup is shown in Fig. 4, including a Raspberry-based control kernel and a mobile robot with Kinect sensor. It is desired to real-time monitor the test environment. The mobile robot can be remotely manipulated via Wi-Fi or Ethernet. Kinect sensor captured images have two functions, RGB color and depth, that will be provided to consequent object detection. Following the design flows addressed in Figs. 1 and 2, the object detection process can be well-performed and the recognition results can be also obtained.

To verify the feasibility of proposed scheme, the following two cases will be discussed:
Case1: Image contains two unique objects
Case2: Image contains one blur object.

The recognition results of Case 1 are shown in Figs. 5 and 6. It can be seen that the captured image contains two unique objects, person and bottle. The detection probability of the person is 99.9%, shown in Fig. 5. On the other hand, the detection probability of the bottle in Fig. 6 is also 99.9%. In Fig. 7, it can be seen that the person in the captured image is not so clear. The person wears a hoodie, and his hand covers most of his face. Following the proposed recognition process, a satisfactory performance sill can be obtained, with the detection probability 99.7%.



(a) **(b)**

Fig. 4 System integration: (a) Raspberry Pi, (b) mobile robot.

ISBN 978-1-5386-4342-6

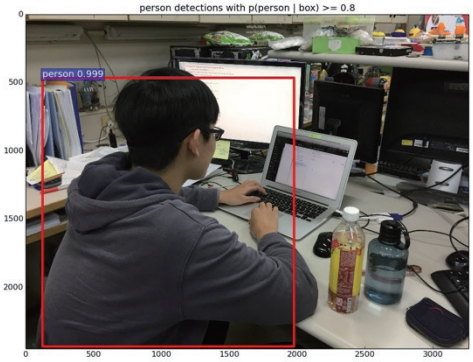


Fig. 5 Object detection (Case 1, person).

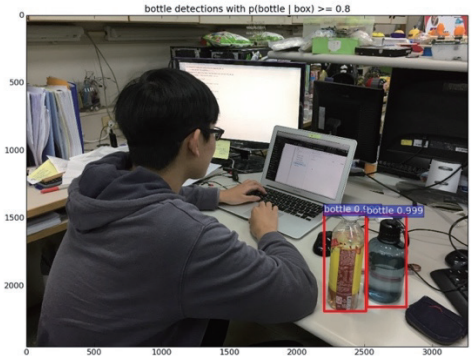


Fig. 6 Object detection (Case 1, bottle).

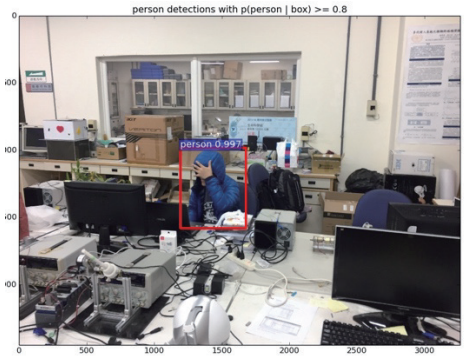


Fig. 7 Object detection (Case 2).

Conclusion

In this work, a Raspberry-based mobile robot is constructed for the purse of object recognition. The ROS middleware is applied to develop the functions such as the movement of mobile robot and the image capturing of Kinect sensor. To enhance the recognition results, Faster R-CNN deep learning algorithm is implemented. To reduce processing runtime, Caffe framework is used to build learning network and integrate with GPU acceleration. In addition, the object detection results are involved in a cloud service platform. Experimental results are provided to verify the feasibility of the proposed scheme. Testing results indicate that the desired objects can be identified with high accuracy.

References

[1] X. Li, B. J. Choi, "Design of Obstacle Avoidance System for Mobile Robot using Fuzzy Logic Systems", International Journal of Smart Home, Vol. 7, No. 3, May, 2013

[2] On S,Yazıcı A., A reference governor approach for path tracking of nonholonomic mobile robot, INISTA 2011, International Symposium on Innovations in Intelligent Systems and Applications, 15-17June 2011, İstanbul Turkey, pp. 550-554.

[3] A. Ashiquzzaman and A. K. Tushar, "Handwritten Arabic numeral recognition using deep learning neural networks," in *2017 IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, 2017, pp. 1-4.

[4] S. Ren K. He R. B. Girshick J, "Sun Faster R-CNN: towards real-time object detection with region proposal networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2015.

[5] B. Liu, W. Zhao, and Q. Sun, "Study of object detection based on Faster R-CNN," in *2017 Chinese Automation Congress (CAC)*, 2017, pp. 6233-6236.

[6] Robot Operating System, http://www.ros.org/

[7] C. J. Baby, F. A. Khan, and J. N. Swathi, "Home automation using IoT and a chatbot using natural language processing," in *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*, 2017, pp. 1-6.

[8] N. J. Uke and P. R. Futane, "Efficient method for detecting and tracking moving objects in video," in 2016 IEEE International Conference on Advances in Electronics, Communication and Computer Technology (ICAECCT), 2016, pp. 343-348.

[9] Everingham M, Winn J, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Development Kit," International Journal of Computer Vision, vol. 111, pp. 98-136, 2015.

[10] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dolla´r, and C. Zitnick. Microsoft COCO: Com- mon Objects in Context. In *ECCV*, 2014.

[11] J. Deng, W. Dong, R. Socher, L. J. Li, L. Kai, and F.-F. Li, "ImageNet: A large-scale hierarchical image database," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255.

[12] V. Bayar, B. Akar, U. Yayan, H. S. Yavuz, and A. Yazici, "Fuzzy logic based design of classical behaviors for mobile robots in ROS middleware," in *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*, 2014, pp. 162-169.

[13] Miao Sun, Tony X. Han, Ming-Chang Liu, Ahmad Khodayari-Rostamabad, "Multiple Instance Learning Convolutional Neural Networks for object recognition", Pattern Recognition (ICPR) 2016 23rd International Conference on, pp. 3270-3275, 2016.

[14] Y. W. Seo, J. Kim, and R. Rajkumar, "Predicting dynamic computational workload of a self-driving car," in 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2014, pp. 3030-3035.

[15] J. Roffey-Barentsen, "The Voices of Teaching Assistants (Are We Value for Money?)", Research in Education, vol. 92, no. 1, pp. 18-31, 2014.

[16] Ankush Gupta, Andrea Vedaldi, Andrew Zisserman, "Synthetic Data for Text Localisation in Natural Images", Computer Vision and Pattern Recognition (CVPR) 2016 IEEE Conference on, pp. 2315-2324, 2016.

[17] Y. Jia et al., "Caffe: Convolutional Architecture for Fast Feature Embedding," presented at the Proceedings of the 22nd ACM international conference on Multimedia, Orlando, Florida, USA, 2014

[18] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on com- puter vision*, pages 818–833. Springer, 2014.

[19] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. Computer Science, 2014.

[20] F. Weninger, J. Bergmann, and B. Schuller, "Introducing CURRENNT: The Munich Open-Source CUDA RecurREnt Neural Network Toolkit," J. Machine Learning Research, vol.16, pp.547-551, 2015.

ISBN 978-1-5386-4342-6