# Automatic Detection Method of Picture-based Lingwu Long Jujubes

Shuai Pang [1, a], Jiangming Kan [2, b] and Yutan Wang [3, c]

[1] School of Electrical Engineering, Binzhou University, Binzhou 256600, China;

[2]School of Technology, Beijing Forestry University, Beijing 100083, China

[3]School of Mechanical Engineering, Ningxia University, Ningxia 750021, China.

[a] psh1988@126.com, [b]Kanjm@bjfu.edu.cn, [c]wancmeil@sina.com

## Abstract

With the continuous development of automation technology, Lingwu Long Jujube picking mechanization becomes the development trend of the future. Object detection is a primary mission in computer vision that focuses on localizing and classifying various objects in an image. In this paper, through experiments, RCNN and Faster RCNN algorithms are used to realize the detection function of picture-based Lingwu long jujubes, and the experimental process and results are analyzed. The results show that the accuracy can be increased by adding the samples in training set by using Faster RCNN.

## Keywords

Object detection; convolutional neural network (CNN); RCNN; Faster RCNN.

## 1. Introduction

Lingwu long jujube is an important economic fruit of the Ningxia Hui Autonomous Region and plays a pivotal role in the local forest fruit economy in Ningxia. Due to environmental factors and its own characteristics, it is difficult for the picking robot to correctly detect and accurately locate the fruits. Object detection is an important topic in the field of computer vision and its main task is to locate the objects of interest in the images, which needs to accurately determine the specific category of each object and give the bounding box of each object [1-3]. In recent years, object detection technology has become increasingly mature and has solved many practical problems in production and life, which has been widely used in various fields such as intelligent video monitoring, vehicle automatic driving, robot environment perception, face recognition and so forth [4-7].

The research on object detection algorithms has been in existence for a long time, and their development has gone through many stages. In simple terms, it can be divided into two major categories: one is traditional algorithms, and the other is algorithms based on deep learning. The current mainstream object detection algorithms based on deep learning mainly have two major branches: one is the RCNN series algorithms based on possible regions, including RCNN algorithm, FastRCNN algorithm and FasterRCNN algorithm [8, 9]; the other is YOLO series of algorithms based on direct regression, among which YOLO algorithm and SSD algorithm are typical [10].

In this paper, Matlab software is used to firstly perform data preprocessing, then RCNN algorithm and Faster RCNN algorithm are implemented. According to the experimental results, improvement measures are proposed to provide technical supports for the automatic detection of Lingwu long jujubes.

## 2. Data Preprocessing

### 2.1 Picture Collection

The pictures used in this research are all taken and collected from the economic forest field of Lingwu long jujubes. There are 3,075 pictures of Lingwu long jujubes selected from them, and each picture contains 0 to several long jujubes, constituting to a dataset. The selected samples contain thousands of different jujube trees, including different growing directions and maturity levels of the jujubes, and the growth-forms under various light and dark illuminations, as well as the actual growth of the jujubes such as the mutual overlap, adhesion, irregular shapes, etc. Some of the collected pictures are shown in Fig. 1.

Fig. 1 Samples in the dataset

### 2.2 Pictures Pre-processing and Labeling

Before the experiment, the collected pictures need to be pre-processed to form a dataset that can be invoked to run in the program.

Since the computation amount of CNN is very large, in order to reduce the amount of computation, shorten the running time of the program, and improve the detection efficiency, it is necessary to program the size of the pictures for unified modification and appropriately reduce the pixel of pictures, and pixel selected here is 228*128. After completing the above work, the image batch processing software Image Tuner is used to uniformly sort and name the pictures, for example: image_00001.jpg, and put them into a folder for backup, as shown in Fig. 2.

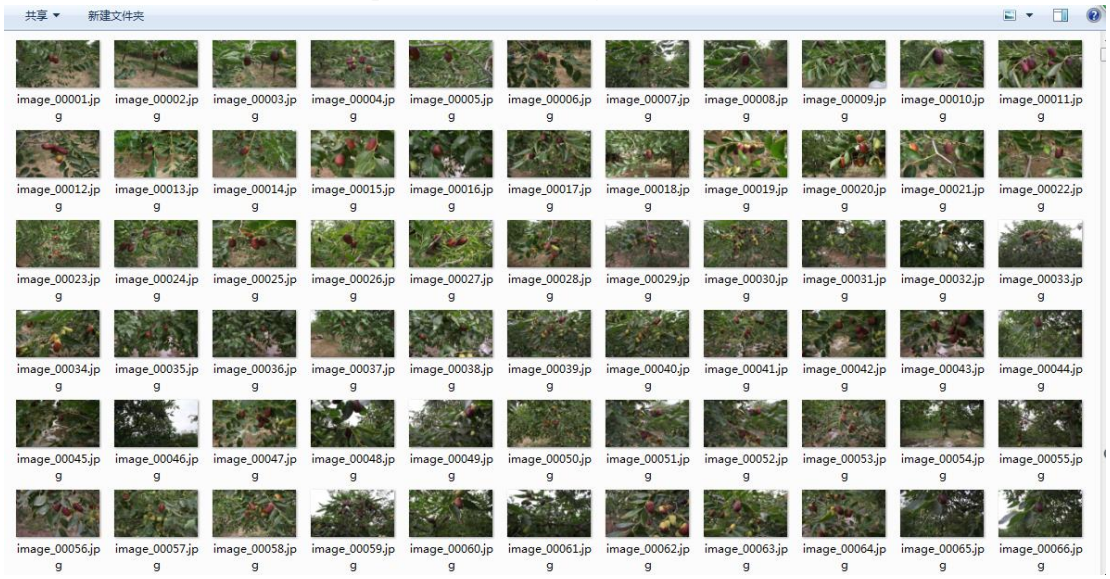


Fig. 2 Pictures to be normalized

After uniformly sorted and named, the pictures can be placed in MATLAB in order and in batches for labeling. MATLAB's built-in image labeling tool Image Labeler is used for pictures labeling, loading the pictures in the dataset into the interface, and adding the label of jujube to label the jujubes in the pictures. The process is shown in Fig. 3.

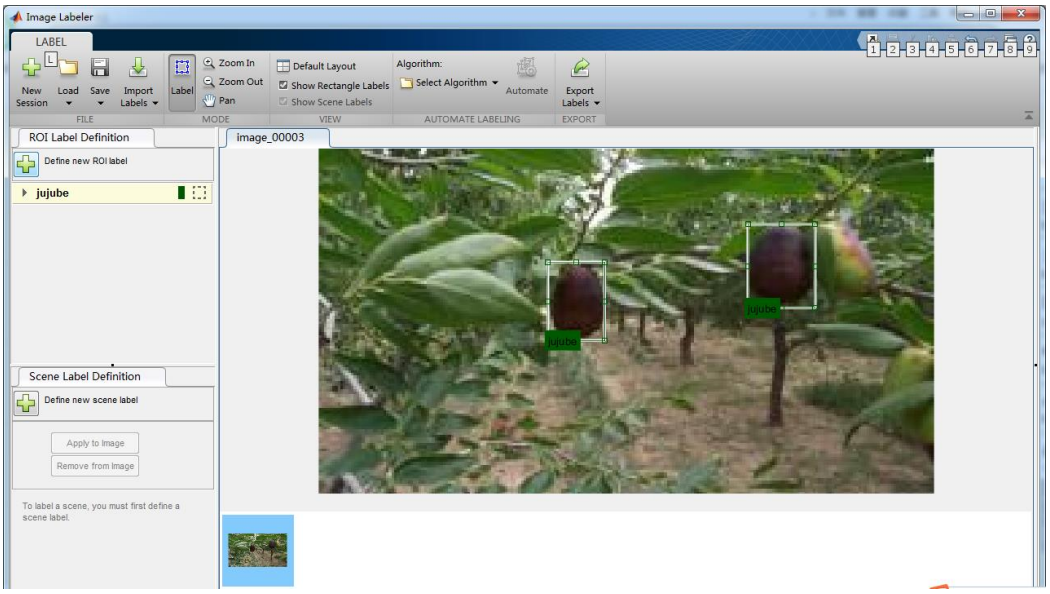


Fig. 3 Image annotation

All the pictures are saved with Export Labels after being labeled, then a table-type.mat file will be generated. As shown in Fig. 4, this is the date set that can be used for CNN training. The first column in the Fig. 4 represents the locations where the pictures are stored, and the second column represents the locations of the long jujubes labeled in the corresponding pictures, expressed in matrix. Each jujube is labeled with four dimensions [x, y, width, height], i.e., the coordinates (x, y) of the upper left corner and the width and height are used to store the positions of the jujubes.

jujube

3075x2 table

| | 1<br>imagefilename | 2<br>jujube |
|---|---|---|
| 1 | 'E:\bishe\photo\final\image_00001.jpg' | [93,40,11,18;104,46,13,20] |
| 2 | 'E:\bishe\photo\final\image_00002.jpg' | [107,31,14,22;126,26,15,19] |
| 3 | 'E:\bishe\photo\final\image_00003.jpg' | [88,45,20,26;161,31,24,27] |
| 4 | 'E:\bishe\photo\final\image_00004.jpg' | *9x4 double* |
| 5 | 'E:\bishe\photo\final\image_00005.jpg' | [1,16,33,38;97,32,24,30] |
| 6 | 'E:\bishe\photo\final\image_00006.jpg' | [80,33,16,17;95,39,20,25] |
| 7 | 'E:\bishe\photo\final\image_00007.jpg' | [91,44,15,22;107,41,16,25] |
| 8 | 'E:\bishe\photo\final\image_00008.jpg' | [103,38,24,36] |
| 9 | 'E:\bishe\photo\final\image_00009.jpg' | *4x4 double* |

Fig. 4 Dataset file

## 3. Implementation of RCNN Algorithm

### 3.1 Process of RCNN Algorithm

The general process of RCNN algorithm can be summarized into four steps: image input → candidate box searching → CNN feature extraction → classification.

### 3.2 Experimental Process and Results

3.2.1 Transfer Learning

In order to improve the training efficiency and save storage space, this paper adopts Transfer Learning during the training of RCNN network and conducts training based on the trained CIFAR-10 data set which contains 10 image categories with a network structure divided into 15 layers.

The CIFAR-10 dataset is a basic and general data set for image recognition in machine learning, including 50,000 color images in 10 categories, such as airplanes, cars, birds, cats, deer, dogs, frogs, horses, boats, trucks and so forth. Some of the training images in this data set are shown in Fig. 5.

Fig. 5 CIFAR-10 data set sample

3.2.2 Training RCNN Network

First, a CNN network should be created for training, which consists of a series of layers and each layer defines a specific computation. A complete network consists of three parts: the input layer, the middle layer and the output layer, each of which also builds several layers as needed. Such layers can be built by using the functionality provided by the Neural Network Toolbox.

In this example, the image input layer, the convolutional layer, the relay linear unit (ReLu) layer, the pooling layer, the fully connected layer, the classification layer, and the output layer are used to create a CNN. The input layer defines the type and size of data that CNN can handle. The middle layer is composed of the convolutional layer, the ReLU (corrected linear unit) and the pooling layer which are the core components of CNN. By repeating these three basic layers, a deeper network will be created, but attention should be paid to control the number of pooling layers to avoid losing useful image information. The output layer is mainly to classify and output the results.

After defining the network architecture, the data can be trained and the training Options function is used to build a network training algorithm. The network training algorithm uses the stochastic gradient descent method (SGDM) with an initial learning rate of 0.001. During the training, the initial learning rate is reduced every 8 cycles (1 cycle is defined as a complete data set passing through the entire training). The training algorithm runs for 40 cycles. After the network algorithm is set up, its operation on the CIFAR-10 data set should be observed, and it can load the data set of Lingwu long jujubes after the operation is normal and use the Transfer Learning method to conduct fine tuning of the network to realize the detection of pictures of Lingwu long jujubes. The trainRCNNObjectDetector.m function is mainly used to train the RCNN object detector, which is built in matlab and can be invoked directly. The syntax format adopted is detector = trainRCNNObjectDetector (groundTruth, network, options). The input of this function is the ground truth table, which contains the marked images of Lingwu long jujubes, the pre-trained CIFAR-10 network and the training options. The training function will

automatically modify the original CIFAR-10 network and change the image classifications from the original 10 categories into two categories: long jujubes and other background. The first column of ground Truth must be the image path and file name, and the remaining columns are the corresponding matrix parameters, each of which represents a single object. The Network is the network structure of CNN, and the options are the parameters of network training, including initial learning rate, iterative times, step size, positive and negative sample parameters, and so on.

3.2.3 Detection Results

The pictures of long jujubes are detected through the above-mentioned operationally trained detector, the locations of long jujubes in the pictures are marked with rectangular boxes and the word "jujube". The typical drawing of results obtained is shown in Fig. 6 below, which can basically meet the expected effects and accurately locate the long jujubes and mark them out.


Fig. 6 RCNN detection results

Fig. 6 shows the detection results for a single picture. The program actually has detected all the pictures of the detection set, but in order to save the running time and storage space of the program, not all the detected graphics are output. However, the overall average precision is obtained through calculation, and the average precision of this experiment is 0.7456.

## 4. Implementation of Faster RCNN Algorithm

### 4.1 Process of Faster RCNN Algorithm

The Faster RCNN algorithm is summarized as the following steps: (1) input the entire picture into CNN to obtain the feature map; (2) input the convolution features into RPN to obtain the feature in formation of the candidate box; (3) use a classifier to determine whether features extracted by the candidate box belong to a particular class; (4) use a regression to further adjust the location of the candidate box belonging to a certain feature.

### 4.2 Experimental Process and Results

4.2.1 Establishment of CNN Network

Similar to RCNN algorithm, the establishment of CNN network is still the most critical part of the implementation of Faster RCNN algorithm. CNN network is the basis for the object identification of CNN, and the neural network toolbox, MATLAB, provides the basic functions for establishing the CNN network, which is also mainly composed of the input layer, the middle layer and the output layer of the network.

First, the input layer of the network is defined, and the type and dimension of CNN network is set through the imageInputLayer function. As there are different kinds of application scenarios, the input dimensions are also varied, among which the input dimension of the object detection application is generally equal to the minimum size of the detection objects; the input dimension of the graphics classification is generally equal to the size of the training graphics. This design is to perform object detection, and the pixel of the smallest long jujube area in the training data is about 32*32, so it is set to 32*32.

Second, the middle layer of the network is defined, which is the core of CNN, consisting of convolution function, activation function, and pooling function. The middle layer can use the convolution function for multiple times, but in order to avoid the loss of image details due to excessive sampling of the image, the number of pooling layers to be used should be minimized. The middle layer of this design first defines the parameters of the convolutional layer, and then uses two-round convolution for calculation. The first round only contains CNN and ReLU, and the second round contains CNN, ReLU, and Pooiing.

Then the output layer of the network is defined, which is generally composed of a classic fully connected layer and a classification layer for the output of the results. In this part, the design adds a fully connected layer containing 64 outputs and a nonlinear ReLU layer behind the middle layer, and a new fully connected layer with 2 outputs is also added to determine whether the image contains detection objects. After that, the soft max and classification layer are added and a output layer is completed.

Finally, the network is defined. A final CNN will be formed by the combination of all the layers and the connection of the input layer, the middle layer and the output layer.

4.2.2 Training and Detection of CNN Network

In order to comprehensively utilize the data, the 3,075 data samples should be reasonably arranged, and the data is divided into training data and test data through programming before training. For this purpose, according to the labels of data, the first 60% of data is used for data training and the remaining 40% is used for detection.

The neural network toolbox provides the train Faster RCNN Object Detector for CNN network training, and the entire training process consists of 4 steps, each of which can specify different training parameters or use the same training parameters. In this design, the learning rate of the first two steps is set to le-5, and the remaining two steps are set to le-6. Fig. 7 presents the partial operation of Step1, and the operations of Step2 to Step4 are similar to that of Step1.

```
*************************************************************************
Training a Faster R-CNN Object Detector for the following object classes:

* jujube

Step 1 of 4: Training a Region Proposal Network (RPN).
Training on single GPU.
|=========================================================================================|
|     Epoch    |   Iteration  | Time Elapsed |  Mini-batch  |  Mini-batch  | Base Learning|
|              |              |   (seconds)  |     Loss     |   Accuracy   |     Rate     |
|=========================================================================================|
|            1 |            1 |         0.34 |       0.6935 |       30.71% |     1.00e-05 |
|            1 |           50 |         5.42 |       0.6889 |       74.22% |     1.00e-05 |
|            1 |          100 |        10.36 |       0.6910 |       66.41% |     1.00e-05 |
|            1 |          150 |        15.36 |       0.6905 |       51.56% |     1.00e-05 |
|            1 |          200 |        20.26 |       0.6787 |       71.88% |     1.00e-05 |
|            1 |          250 |        25.16 |       0.6656 |       75.00% |     1.00e-05 |
|            1 |          300 |        30.11 |       0.5712 |       69.53% |     1.00e-05 |
|            1 |          350 |        35.02 |       0.5221 |       73.44% |     1.00e-05 |
|            1 |          400 |        39.95 |       0.5517 |       67.97% |     1.00e-05 |
|            1 |          450 |        44.77 |       0.6411 |       83.59% |     1.00e-05 |
|            1 |          500 |        49.75 |       0.4187 |       88.28% |     1.00e-05 |
|            1 |          550 |        54.61 |       0.3689 |       97.66% |     1.00e-05 |
|            1 |          600 |        59.49 |       0.4146 |       81.25% |     1.00e-05 |
|            1 |          650 |        64.37 |       0.4891 |       75.78% |     1.00e-05 |
|            1 |          700 |        69.26 |       0.2875 |       93.75% |     1.00e-05 |
|            1 |          750 |        74.14 |       1.1167 |       57.81% |     1.00e-05 |
|            1 |          800 |        79.05 |       0.3754 |       87.50% |     1.00e-05 |
|            1 |          850 |        83.96 |       0.2763 |       88.28% |     1.00e-05 |
|            1 |          900 |        88.89 |       0.3390 |       88.28% |     1.00e-05 |
|            1 |          950 |        93.77 |       0.2194 |       95.31% |     1.00e-05 |
|            1 |         1000 |        98.69 |       0.1632 |       96.09% |     1.00e-05 |
|            1 |         1050 |       103.58 |       0.1113 |      100.00% |     1.00e-05 |
|            1 |         1100 |       108.42 |       0.8385 |       50.00% |     1.00e-05 |
|            1 |         1150 |       113.35 |       0.3710 |       86.72% |     1.00e-05 |
|            1 |         1200 |       118.21 |       0.5736 |       73.44% |     1.00e-05 |
|            1 |         1250 |       123.16 |       0.1506 |       98.44% |     1.00e-05 |
```

Fig. 7 Step1 run the process

Before training, it needs to ensure that the data set has been loaded in the program, and in order to visualize this process, a piece of code is programmed for the program to output 9 sample images after the successful extraction of data. The operating result of this process is shown in Fig.8 which shows the first 9 images and adds the labels of the matrix boxes of the objects.

Fig. 8 The first nine sample images

After a certain period of training, the model of CNN network is obtained. For accelerating the detection, a picture is selected for input, and the detector is operated to output the locations and scores of the objects, which indicates that the model of CNN network can successfully detect the long jujubes and show the labeling locations. The experiment shows that when the number of pictures in the training set is different, the effect of detection will also have a little difference. Fig. 9 presents the result of a training set of 100 pictures, while Fig. 10 presents the result of a training set of 1845 pictures. Apparently, the more pictures are trained, the more accurate the location boxes will be, and the better the detection effect will be.

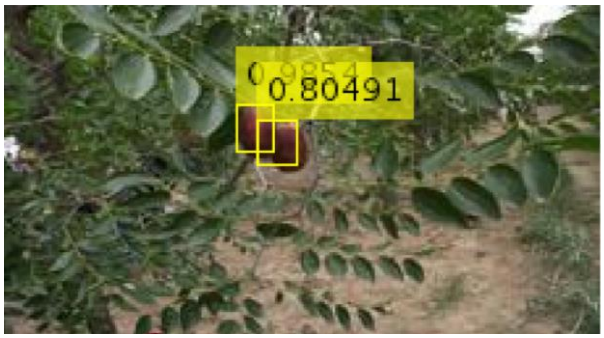


Fig. 9 The training results for 100 samples


Fig.10 The training results for 1845 samples

4.2.3 Evaluation of Training Effect

Among the fast detection results shown in Fig. 9 and Fig. 10, the numbers above the location boxes of the objects, such as 0.92212 and 0.97615, refer to the scores of the object boxes, which are the positional accuracy and a visual evaluation of the detection effect.

The detection results shown in Fig. 10 are fair good, and it can be seen that the detector has accurately located the long jujubes' locations in the picture and the scores of object boxes are above 0.95, which achieves the desired effect. However, this is only a fast detection result of a single image which does not represent the entire data set. So in order to better verify the training effect of CNN, it is necessary

to conduct a large-scale detection. MATLAB, the computer visual toolbox, provides an average precision evaluationDetectionPrecision function and a logarithm-average error rate evaluation Detection Miss Rate function to evaluate the detector's training effect. This design adopts the evaluationDetectionPrecision function to evaluate and calculate the recall rate and precision rate as the evaluation criteria.

In this design, a set of 100 pictures is first detected, and the average precision rate is 0.09. After the 100 samples are successfully detected, the samples are increased, and a set of 3,075 pictures are evaluated with an average precision rate of 0.49.

From the comparison of the two results, increasing the number of training samples will improve the average precision rate of the detection. Therefore, when training the detector, a large number of training samples are one of the key factors to ensure the quality of detection. In addition, it's considerable to try to improve the precision by increasing the number of CNN network layers, but the cost of training and detection will be increased at the same time.

## 5. Conclusion and Prospects

In terms of usage, RCNN extracts candidate regions with selective searching, extracts features with CNN, classifies them with SVM classifier, and uses bounding box to perform regression; while FasterRCNN extracts candidate regions with RPN network, extracts the features with CNN, classifies them with softmax, and uses the multi-task loss function to regress. Regarding the disadvantages, the RCNN algorithm is cumbersome, the speeds of training and detection are slow, and the training space is very large; although the FasterRCNN still cannot monitor the objects in real time, it has simplified a lot of steps based on RCNN and has greatly improved the speeds of training and detection. Due to the reduction of the computation, the occupied space has also been greatly improved. The end-to-end object detection framework is realized, and the generation of the suggesting box takes only 10 ms, which greatly improves the precision and speed of detection.

It can be seen from the detection results that the detection effect is good for pictures with simple background and few jujubes, but for pictures with complicated background and a large number of jujubes, false detection and multiple repeated detection windows are likely to occur; the background objects such as leaves that is similar to long jujubes in appearance and colors are also easily detected; for some pictures, only part of a jujube can be detected, but not the entire one; the farther and the smaller the jujubes are, the higher the false detection rate will be.

The following methods are for improving the above-mentioned defects in the detection results:

1. Increase the number and diversity of training samples and the times of training, so that the network can better learn the characteristics of long jujubes. The difference in experimental results between the training samples of 100 and 3075 pictures also verifies well that a large number of training samples can directly and effectively improve the detection precision.

2. Improve the CNN network into a structure with better generalization ability to increase the recognition rate. For example, the network structure of CNN can be improved by increasing the number of convolution layers and adjusting the parameters of the convolution layers.

3. For selecting the training samples, priority should be given to the pictures with fewer, less dense, and complete jujubes. This is because that during picture training, the values of PositiveOverlapRange, NegativeOverlapRange parameters are selected to obtain positive training samples and negative training samples. If the jujubes are dense and there are many obstructions, it is difficult to accurately mark all the jujubes, and it is easy to handle the long jujubes as negative samples during training. Meanwhile, several jujubes may be marked with one box during the detection.

## Acknowledgements

This work is supported by the National Natural Science Foundation of China (31660239), the Ningxia Natural Science Foundation of China (NZ15007) and Binzhou University Doctoral Research Program (2016Y19).

## References

[1] M.J. Zhang, V.B. Ciesielski and P. Andreae: A Domain-Independent Window Approach to Multiclass Object Detection Using Genetic Programming, EURASIP Journal on Advances in Signal Processing, vol. 8 (2003), p.841-859.

[2] M. Lokanath, K.S. Kumar and E.S. Keerthi: Accurate object classification and detection by faster-RCNN, IOP Conference Series: Materials Science and Engineering, Vol.263 (2017) No.5, p. 052028.

[3] T. Kryjak, M. Komorkiewicz and M. Gorgon: Real-time Foreground Object Detection Combining the PBAS Background Modelling Algorithm and Feedback from Scene Analysis Module, International Journal of Electronics and Telecommunications, Vol.60 (2014) No.1, p. 53-64.

[4] S. Yoshida, H. Okada, T. Ogawa, et al. A Method for Improving SVM-based Image Classification Performance Based on a Target Object Detection Scheme, ITE Transactions on Media Technology and Applications, Vol.1 (2013) No.3, p. 237-243.

[5] A. Lorencs, I. Mednieks and J.S. Siņavskis: Fast object detection in digital grayscale images, Proceedings of the Latvian Academy of Sciences. Section B. Natural, Exact, and Applied Sciences, Vol.63 (2009) No.3, p. 116-124.

[6] K. Takagi, K. Tanaka, S. Izumi, et al. A Real-time Scalable Object Detection System using Low-power HOG Accelerator VLSI, Journal of Signal Processing Systems, Vol.76 (2014) No.3, p. 261-274.

[7] J. Gance, J. P. Malet, T. Dewez, et al. Target Detection and Tracking of moving objects for characterizing landslide displacements from time-lapse terrestrial optical images, Engineering Geology, Vol.172 (2014), p. 26-40.

[8] Y. Chen, D. Zhao, L. Lv, et al. Multi-task learning for dangerous object detection in autonomous driving, Information Sciences, Vol.432 (2018), p. 559-571.

[9] M. Turan,Y. Almalioglu, H. Araujo, et al. Deep EndoVO: A Recurrent Convolutional Neural Network (RCNN) based Visual Odometry Approach for Endoscopic Capsule Robots, Neurocomputing, Vol.275 (2018), p.1861-1870.

[10] P. Christiansen, L.N. Nielsen, K.A. Steen, et al. DeepAnomaly: Combining background subtraction and deep learning for detecting obstacles and anomalies in an agricultural field, Sensors, Vol.16 (2016) No.11, p.1904.