Development and testing of a conveyor for detecting various types of vehicles when transporting agricultural products from the field

I A Uspenskiy, G K Rembalovich, I A Yukhin, D S Ryachikov and A S Stepashkina

Ryazan State Agrotechnical University named after P.A. Kostychev, 1, Kostycheva Str., Ryazan, 390044, Russia

E-mail: ivan.uspensckij@yandex.ru  rgk.rgatu@yandex.ru  rds_62@mail.ru

Abstract. Research purpose: development of a technology that will detect various types of vehicles in the image. To achieve this purpose it is necessary to solve the following objectives: Highlighting requirements for the technology being developed; Development of the technology for finding an object in an image; The choice of methods and algorithms for the allocation of objects for the development of the technology; The choice of methods and algorithms that allow detecting objects of a certain class; Implementation of the developed technology in the software system. The scientific novelty of the work lies in the application of previously known methods, which have been shown to be effective in other studies, to a new object of study, namely, to detect in images vehicles of various classes used for the transportation of agricultural goods. The practical value of the expected results lies in creating the software based on the developed technology, which makes it possible to detect vehicles of a certain class with a high degree of probability in a static image. The development of such software will automate part of the business processes and reduce labor costs. The following methods are used as the main ones in the developed technology: the HOG method (histograms of oriented gradients) and the support vector method (SVM).

1. Introduction
In modern conditions of technology development, people are increasingly striving to automate processes. Automation is understood as a process where previously performed functions are transferred by a person to some equipment and automatic devices. Automation surrounds us everywhere: from domestic to world-class issues.

Undoubtedly, high-tech devices have already been developed and are widely used, which make it possible to collect the necessary data on the parameters of the movement of vehicles from harvesting sites to storage units. For the operational management of freight transport during loading, it is necessary to know its carrying capacity in advance. Determining the class of the vehicle and its load capacity in an automated way will reduce human labor costs. However, all existing special instruments for measuring the intensity of the traffic flow have a high price. Therefore, in practice, the manual method of determining the class of vehicles is often used.

To reduce capital and labor costs, a decision was made to develop a technology for detecting various types of vehicles in the image [1].
The task of finding an object in an image is complicated by a number of factors: distortion of an object in an image, overlapping of objects with other objects, stray signals in the image, a variety of objects in the shape and color of objects of the same class, etc.

When creating automated systems several widespread approaches to search for objects in the image are used.

Depending on the method of obtaining the source image for solving the problem of finding objects, the following approaches are used.

Modeling the background. This approach is applied when a stable and slightly changing background, i.e., an image comes from a fixed camera.

Modeling an object. The approach is applied in cases where the background regularly and substantially changes. To solve some problems, it is appropriate to apply both approaches together, this can significantly improve the results.

2. The choice of the sequence of the main stages of the developed technology

The work of Andrey Vedaldi and Andrew Zisserman was chosen as a model for the technology to be developed [2].

In their work, the researchers set a goal: the discovery of objects of a certain category. Pedestrians, cars, traffic signs on the city streets were identified as search objects.

The source data for the program are images that depict the target search objects. At the output of the program the target objects are marked with a bounding box.

The problem of finding an object in an image is the search for the object by the detector, regardless of the location of the object, its scale in the image, as well as lighting, color, etc.

The search for objects in the image in the technology developed by the researchers is to search for objects in a training sample. That is, before detecting an object in the image, a statistical analysis of the image is performed. And then, to detect the object in the image, the obtained model of the object is applied with the help of a sliding window to all parts of the image at all possible scales [3].

Now let's pass to a more detailed study of the developed technology.

3. Basics of detection

The target objects of the search are vehicles of two classes: cars and trucks.

Two classes of cars were chosen as the initial data for training, and street pictures with cars were selected as test images.

To search for objects in the test image, the sliding window method will be used, based on the construction of the HOG feature extraction method (histograms of directed gradients).

In MATLAB, a training sample for practical use is loaded with the m-file loadData.m. The loadData (targetClass) function takes a targetClass argument that points to an object of the interest class.

The LOADDATA function loads data for training. LOADDATA (TARGETCLASS) loads data, configuring it for training. TARGETCLASS - the target class - is the vector of one or more labels. If several labels are specified, then several classes are combined into one. LOADDATA (TARGETCLASS, NUMPOSIMAGES, NUMNEGIMAGES) indicates positive and negative examples.

Figure 1 shows the elements of the training sample related to two classes of objects. On the left side of each figure there are examples of objects of one class (type) of cars and on the right side of each figure there is an average image of cars of one class.

Consider the first part of the program.

When working out the example1.m script, variables are loaded into the working space:

- trainImages – a list of training sample image names;
- trainBoxes – 4 × N array variable of bounding rectangles having coordinates \([x_{min}, y_{min}, x_{max}, y_{max}]\);
- trainBoxImages – the name of the object enclosed in the bounding box;
- `trainBoxLabels` – the label of the object for each bounding box (one of the indices in target class `targetClass`);
- `trainBoxPatches` – an array variable of image sections, one for each training object (in RGB format).

![Figure 1. Elements of the training set of cars of two classes](image1.png)

A similar set of `testImages`, `testBoxes` variables was created for checking test images.

In step 1.1, an image is created that visualizes the complete training list of objects of one class and their average value.

Figure 2 shows the image obtained in step 1.1. The figure shows cars. All elements belong to the same class, on the right side of the image there is an average image obtained from the sample elements of this class.

![Figure 2. The image visualizing the complete list of training for objects of one class (cars) and their average value](image2.png)
The same actions are performed to obtain an average d value for a training sample consisting of trucks (Figure 3).

![Figure 3](image-url)

**Figure 3.** The image visualizing the complete training list of objects of one class (trucks) and their average value

### 4. Extracting HOG features from training images

In most cases, detectors work on top of a layer of low-level objects. In this case, a HOG detector (histograms of oriented gradients) is used. To obtain a HOG model of an object, objects are extracted from parts of the images corresponding to the existing training examples. This action is performed in the next cycle:

```matlab
hogCellSize = 8;
trainHog = {};
for i = 1:size(trainBoxPatches,4)
    trainHog{i} = vl_hog(trainBoxPatches(:,:,:,:,i), hogCellSize);
end
trainHog = cat(4, trainHog{:});
```

HOG is computed by the VLFeat `vl_hog` (doc) function. This function takes as a parameter the pixel size for each HOG cell. It also accepts an RGB image represented in MATLAB as a \( w \times h \times 3 \) array variable (extracted as a fragment of trainBoxPatches patches). The output is the size of the array variable: \( w / \text{hogCellSize} \times h / \text{hogCellSize} \times 31 \). One array variable is extracted for each training sample, then the array variables are combined to form a 4D array variable (fourth dimension).

### 5. Building a model of the HOG pattern

A simple object model can be obtained by averaging the characteristics of the objects in the training sample.

This is done in the following way:

```matlab
w = mean(trainHog, 4),
```

where \( w \) – the average value of HOG.
The model can be visualized by rendering $w$, like an array variable of HOG objects. Rendering (Engl. Rendering - “visualization”) is a term used in computer graphics, denoting the process of obtaining images from a model using a computer program.

In this case, the operation is performed using render option `vl_hog`:

```matlab
figure(2) ; clf ;
imagesc(vl_hog('render', w)) ;
```

The result of the program during this step is shown in Figures 4 (a car) and 5 (a truck).

![Figure 4](image-url)  
**Figure 4.** The result of building a model of the HOG pattern of a car

![Figure 5](image-url)  
**Figure 5.** The result of building a model of the HOG pattern of a truck
6. Applying the HOG model to a test image
The model is compared with the test image by: extracting the HOG model of the image and convolution of the model according to the resulting map of objects:

```matlab
im = imread('data/test.jpg');
im = im2single(im);
hog = vl_hog(im, hogCellSize);
scores = vl_nnconv(hog, w, []);
```

The first two lines read an example image and convert it to one format. The third line computes the HOG model using `vl_hog` described above. In the fourth line, the HOG convolution from pattern `w` is compared with the HOG model of the test image, returning a response map.

The result of the program at this step is shown in Figures 6 and 7.

![Figure 6. Car image response map](image1)

![Figure 7. Truck image response map](image2)
On this map, the image area is highlighted in yellow, where the HOG model of the training sample examples is most identical to the HOG model of the test image area, the opposite value is shown in blue.

7. Selecting an object in the image

After receiving the system response map, the image area is found where the maximum response is observed and the bounding box of the image section containing the corresponding HOG model is calculated.

The maximum response is:

\[
[\text{best}, \text{bestIndex}] = \max(\text{scores}(::));
\]

It is worth noting that bestIndex is a linear index in the range \([1, M]\), where \(M\) is the number of possible filter locations.

The lower left coordinate of the bounding box \((h_x, h_y)\) is calculated in MATLAB using function ind2sub:

\[
[h_y, h_x] = \text{ind2sub}(\text{size}(\text{scores}), \text{bestIndex}).
\]

Coordinates \((h_x, h_y)\) are calculated in units of HOG cell values. Converting values to pixel coordinates is as follows:

\[
x = (h_x - 1) \times \text{hogCellSize} + 1;
\]

\[
y = (h_y - 1) \times \text{hogCellSize} + 1.
\]

The size of the model pattern in the coordinates of the HOG cells can be calculated in several ways and one of them is as it follows:

\[
\text{modelWidth} = \text{size(trainHog, 2)};
\]

\[
\text{modelHeight} = \text{size(trainHog, 1)}.
\]

Thus, one gets enough information to calculate the bounding box:

\[
detection = [x - 0.5;
y - 0.5;
x + \text{hogCellSize} \times \text{modelWidth} - 0.5;
y + \text{hogCellSize} \times \text{modelHeight} - 0.5].
\]

A bounding box spans a portion of the image including the exact size of the HOG model. In MATLAB, pixel centers have integer coordinates, and the pixel borders are ±0.5.

According to the results of step 1.5, the area of the vehicle of one of the classes is selected on the test image as a bounding box of rectangles (Figures 8 and 9).

The result of the program is an object enclosed in a bounding box on the test image. However, the search for the object is carried out correctly, provided that the size of the object in the image matches the size of the HOG model, that is, it does not scale.

8. Image scalability and application of SVM reference vector methods

The implementation result in the first part of the program gave a positive result, however, the algorithm described above does not work well when changing the scale of an object in the image.

In the second part, we expand the detector for searching for objects in cases of zooming and improve the algorithm for searching for an object in the image by using the support vector method. To get started, data for cars will be uploaded:
N = 10; % Number of images

trainBoxPatches = single(zeros(64,64,3,N));
for i = 1:N
    trainBoxPatches(:,:,i) = ...   imread(['data/LittleCars/Car00' int2str(i - 1) '.jpg']);
end

**Figure 8.** The result of detecting a car in the image using the construction of the HOG model

**Figure 9.** The result of detecting a truck in the image using the construction of the HOG model
For trucks:
N = 6; % Number of images

\[
\text{trainBoxPatches} = \text{single}(\text{zeros}(64,64,3,N));
\]
\[
\text{for } i = 1:N
\text{trainBoxPatches}(::,:,:i) = \ldots
\text{imread(['data/Truck/truck00' int2str(i - 1) '.jpg'])};
\text{end}
\]

Just as in the example described above, a model of the HOG pattern (simple model) is built. Since the models will be identical to the models built in the example above (Figures 12 and 13), they are not given.

9. Multiscale detection
In most cases, the size of the objects in the image differs from the size of the desired pattern. To solve this problem, the image is scaled in all directions and objects are searched continuously.

\[
\text{% Scale space configuration}
\text{minScale} = -1 ;
\text{maxScale} = 3 ;
\text{numOctaveSubdivisions} = 3 ;
\text{scales} = 2.^\text{linspace(...}
\text{minScale,...}
\text{maxScale,...}
\text{numOctaveSubdivisions*(maxScale-minScale+1))}.
\]

Given \( w \) model defined in part 1 and using detectAtMultipleScales function, objects are searched on several scales:

\[
\text{detection} = \text{detectAtMultipleScales}(\text{im}, \ w, \ \text{hogCellSize}, \ \text{scales}).
\]

As in Part 1, based on a comparison of the HOG convolution from \( w \) pattern and the HOG model of the test image, a response map is constructed at different scales (Figure 10).

**Figure 10.** Making a response map
Based on the comparison of HOG models, objects are searched on the test image. The result of the program at this stage is shown in Figure 11.

![Figure 11. Simple detector output](image)

**Figure 11. Simple detector output**

As a result of the operation of a simple detector, the image area is highlighted on the test image, where in fact the desired object is located.

10. Acquisition of positive and negative training results

Now that there is positive and negative data, you can apply the SVM model for the developed algorithm. Function `vl_svmtrain` will be used for this.

The function requires that the data be combined into a matrix of dimension $D \times N$, where $D$ is the size of the object, and $N$ is the number of training points, i.e.:

```matlab
% Pack the data into a matrix with one datum per column
x = cat(4, pos, neg);
x = reshape(x, [], numPos + numNeg).
```

A vector of binary labels is constructed, where value +1 is assigned for positive points and –1 for negative points:

```matlab
% Create a vector of binary labels
y = [ones(1, size(pos, 4)) -ones(1, size(neg, 4))].
```

Set the value of parameter $\lambda$ for the SVM algorithm, and use it instead of equivalent parameter $C$:

```matlab
numPos = size(pos, 4);
numNeg = size(neg, 4);
C = 10;
lambda = 1 / (C * (numPos + numNeg)).
```

The construction of the HOG model is carried out using the SVM reference vector method (Fig. 12).
Figure 12. Building a HOG model using the SVG support vector method

Again, based on a comparison of the HOG convolution from the pattern and the HOG model of the test image, a response map is constructed at different scales (Figure 13). And the search for objects in the test image is performed (Figure 14 for the search for a car and Figure 15 for the search for a truck).

It can be seen from the results of the algorithm that the joint use of two methods (the oriented gradient method and the support vector method) makes possible to obtain more accurate results of determining the desired objects in the image at different scale values.

From the above example, one can see that the detector recognizes objects in those parts of the image where in fact they do not exist. Therefore, it is necessary to assess the adequacy of the detector. For this, the Pascal criterion VOC with the calculation of the average accuracy value (AP) is used.
Multiple testing of the developed technology made it possible to establish that even with a small training sample, the accuracy of detection of objects of a certain class is quite accurate.

In most cases, objects located in the foreground, i.e. had a greater system response, were detected. The use of only HOG method does not allow to detect objects in the image accurately, as with the simultaneous use of two methods (SVM + HOG). The combined use of two methods shows high efficiency, since it describes the parameters of objects more accurately.

**Figure 14.** The result of the object search using the oriented gradient method (HOG) and the support vector method (SVM) for the search for cars

**Figure 15.** The result of the object search using the oriented gradient method (HOG) and the support vector method (SVM) for the search for trucks
11. Conclusion

Based on the results of this work, the task of developing a technology for detecting various types of vehicles used for the transportation of agricultural goods in a transport stream using images was solved. The practical implementation of the technology made it possible to draw conclusions about the effectiveness of the developed technology, which allows it to work correctly even with a small training sample. The task set has been completely solved: according to the results of the work, a software product was implemented and its effectiveness was confirmed by tests.

Based on the results of the work, the following conclusions are made:

1. The technology for detecting various types of vehicles (trucks and cars) in a static image has been developed.

2. The developed technology for searching for objects in an image using the calculation of the HOG descriptor has shown its effectiveness only in images where the overall dimensions (in pixels) of the desired object coincide with the dimensions of the constructed HOG model based on the elements of the training sample.

3. The developed technology for searching for objects in the image using two methods (oriented gradients and reference vectors) has shown great efficiency in detecting the desired objects. This result was achieved due to the multiple scaling of the image and the search for the object due to the sliding window.

4. The developed technology makes possible to satisfy the requirements for the detection system of an object in the image.

References

[1] Viola P 2001 Rapid Object Detection using a Boosted Cascade of Simple Features Accepted Conf. on Computer Vision and Pattern Recognit. (CVPR 2001) pp 511–8

[2] Object category detection practical Retrieved from: http://www.robots.ox.ac.uk/~vgg/practicals/category-detection/index.html#part2.

[3] Hartley Richard and Zisserman Andrew 2004 Multiple View Geometry in Computer (Vision Cambridge University Press)