Detection of grapes in natural environment using HOG features in low resolution images

Pavel Škrabánek and Filip Majerík

Faculty of Electrical Engineering and Informatics, University of Pardubice, Studentská 95, 532 10 Pardubice, Czech Republic

pavel.skrabanek@upce.cz

Abstract. Detection of grapes in real-life images has importance in various viticulture applications. A grape detector based on an SVM classifier, in combination with a HOG descriptor, has proven to be very efficient in detection of white varieties in high-resolution images. Nevertheless, the high time complexity of such utilization was not suitable for its real-time applications, even when a detector of a simplified structure was used. Thus, we examined possibilities of the simplified version application on images of lower resolutions. For this purpose, we designed a method aimed at search for a detector's setting which gives the best time complexity vs. performance ratio. In order to provide precise evaluation results, we formed new extended datasets. We discovered that even applied on low-resolution images, the simplified detector, with an appropriate setting of all tuneable parameters, was competitive with other state of the art solutions. We concluded that the detector is qualified for real-time detection of grapes in real-life images.

1. Introduction

The latest findings in computer vision were implemented in many agricultural applications and viticulture was no exception. One of the key applications within viticulture was detection of grapes in real scene images. This issue was important in various viticulture applications, such as in autonomous vineyard sprayers, harvesters, or in the process of yield estimation [1-4]. Various image processing, feature extraction and classification algorithms were employed by detection of berries or bunches of grapes in RGB images.

A bunch detector, introduced by Reis at el., utilized colour mapping, morphological dilation, and stem detection [5]. The detector showed correct white wine bunch classification at 90.53 % and red wine at 97.14 %. These results well demonstrated the complexity of white variety detection. Indeed, other solutions aimed at detection of white varieties reached similar results. Berenstein's detector, based on a decision tree algorithm, showed the detection rate of bunches at 90.45 %, and the detection rate of single grapes at 90.10 % [1]. Nuske's solution used a radial symmetry transformation, Gabor filters, and a *k*-nearest neighbours classifier [2]. The detector showed overall precision at 98.00 %, but recall was at only 63.70 %.

A single grape detector, based on the support vector machines (SVMs) classifier, in combination with histograms of oriented gradients (HOG), showed the best performance in white wine varieties detection [6]. Its average accuracy by 10-fold cross-validation (CV) was at 98.23 % for a linear, and at 98.96 % for a radial basis function (RBF) kernel. Moreover, its average precision was at 97.80 % and at 98.49 % for the linear and the RBF kernel, respectively. Its average recall was at 98.68 % and at 99.44 %, respectively. Similar results were achieved by its evaluation on real-life images [7].

Despite the excellent results, the practical utilization of this detector was limited. Indeed, the detector was tested on high-resolution images. Considering the limited computing power of a commonly used hardware in commercial solutions, the detector in its original form could not be used for real-time applications. In order to reduce its time complexity, the structure of the detector was pruned, which resulted in a simplified version of the detector [8]. However, the reduction of the complexity was still insufficient for its use in real-time applications. Here, we address the time complexity vs. performance issue in the context of the image resolution and setting of the detector. We showed that the simplified version of the detector with appropriate setting, guarantees performance comparable with other state of the art solutions.

2. Original works on grape detector

The presented study builds on our previous research published in [6, 8]. Herein, we provide a brief summary of the published detector and datasets.

2.1. Objective of grape detector

The detector was aimed at recognition of grapes of white varieties in object images. RGB object images I (square viewports of source images) of dimensions 40×40 px were considered in [6-8], where px was an abbreviation for a pixel. The resolution of source images was 1936×1288 px, 24 bit.

The detector distinguished between two classes y: 'berry' and 'not berry'. The class 'berry' was called 'positive' and the class 'not berry' was called 'negative'. An object image belonging to the class 'positive' contained a berry of circle shape of diameter ranging between 30 and 40 px. Moreover, the middle of the berry was required to be placed in the middle of the object image with tolerance ±1 px. Object images, which did not satisfy this condition, belonged to the class 'negative'.

2.2. Structure of grape detector

Detection of objects in images usually consists of four successive steps: acquisition of an object image, image pre-processing (IP); extraction of features; and classification. Since the acquisition of object images was not considered in [6, 8], figure 1 shows the complete vision pipeline of the grape detector. Parts of the detector are described in the context of our previous works in further details.

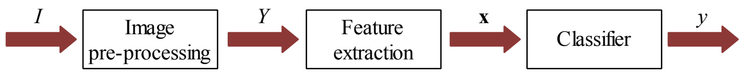

Figure 1. Vision pipeline of grape detectors.

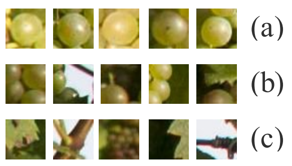

Figure 2. Sample's examples of type: (a) 'positive'; (b) 'negative' in GX, and UX; (c) 'negative' in EX, and UX.

2.2.1. Image pre-processing. Three versions of IPs were introduced. We discovered that a conversion of the input RGB image I to grayscale format Y is the only rational operation to be performed within the IP [8]. The conversion was carried out according to ITU-R recommendation BT.601. This version of the grape detector was named the simplified version.

2.2.2. Feature extraction. The HOG features proved to be convenient for the detection of grapes of white varieties [6]. The feature vector **x** was extracted from Y using a HOG descriptor [9]. We used a conservative setting of the detector [6, 8]: linear gradient voting into 9 bins linearly spread over 0 to

180 degrees; cells of size 6×6 px blocks of 2×2 cells; and 1 overlapping cell between adjacent blocks in both directions.

2.2.3. Classifier. Solutions introduced in [6, 8] used SVMs [10] as the classifier. The linear and RBF kernel functions were considered. Regardless of the used kernel function, performance of the classifiers might be influenced by regularization constant C. Performance of the classifier with RBF kernel was predetermined also by a kernel width σ. A grid search algorithm [11] combined with the 10-fold CV was used to find a setting of these parameters giving maximal accuracy [6].

2.3. Training and evaluation

A training set (T-3) of 288 unique 'positive' and 288 unique 'negative' samples was used for the training of the detector [6, 8]. Three evaluation methods were used for the evaluation of the detector [6-8]; however, only the evaluation on test sets is relevant in the context of this paper.

Two types of sets, five sets of each type, were used for the evaluation [6]. The difference between these two types consisted of selection of the 'negative' samples. While 'negative' samples in G (grape type) were composed solely of incomplete berries of diameter between 30 and 40 px, the 'negative' samples in E (environment type) did not capture even the smallest piece of targeted berries.

50 unique 'positive' and 200 unique 'negative' samples were acquired to create a test set. Each test set was extended by artificial 'positive' samples [12] where these samples were created by turning of the images through an angle φ, i.e. the 'positive' samples were turned by $\varphi \in \{0, \pi/2, \pi, 3\pi/2\}$. It means that each test set consisted of 200 'positive' and 200 'negative' samples. All test sets were based on one vineyard row photo which was not used for creating of the training set T-3.

Three performance measures m (acc – accuracy, pr –precision, re – recall) were used in evaluation of the detectors:

$$\mathrm{acc} = \frac{|TP| + |TN|}{|TP| + |FN| + |TN| + |FP|}, \quad \mathrm{pr} = \frac{|TP|}{|TP| + |FP|}, \quad \mathrm{re} = \frac{|TP|}{|TP| + |FN|}, \tag{1}$$

where $|TP|$ is the number of correctly classified 'positive' samples, $|FN|$ is the number of misclassified 'positive' samples, $|FP|$ is the number of misclassified 'negative' samples, and $|TN|$ is the number of correctly classified 'negative' samples [13].

3. Detector for low–resolution images

In image recognition systems, reduction of the object image resolution brings about a reduction of time-complexity, but often at the expense of the performance. Moreover, the change of the resolution may require significant changes in the image recognition system setting. We addressed all these issues while adapting the grape detector for low-resolution images.

3.1. HOG descriptor properties

Several versions of the HOG descriptor were introduced [9]. The used implementation of the descriptor supports rectangular cells, R-HOG, and spreading of histogram channels over 0 to 180 degrees (unsigned orientation) or 0 to 360 degrees (signed orientation). For the contrast normalization, a L2-hys norm was used. The descriptor has five tuneable parameters: range r (in degrees), number of bins b, size of cells cs (in pixels), number of cells in blocks cb (in cells), and number of overlapping cells between adjacent blocks oc (in cells).

For the used implementation, the range r is a two-valued variable where $r \in \{[0,180],[0,360]\}$ and the number of bins b can be any positive integer. A higher number of bins may ensure encoding finer orientation details, but increasing this value enlarges the feature vector $\mathbf{x}$. Using the signed orientation can help differentiate light-to-dark vs. dark-to-light transitions; however, a higher number of bins b is often needed in order to exploit the potential of the full degrees range.

The remaining three parameters, cs, cb, and oc, are each defined by two values where the first value relates to vertical direction (subscript V) while the second one to horizontal (subscript H). The values can be any positive integer (*oc* can contain also zeros) up to limitations following from image resolutions. Increasing the cell size *cs* or block size *cb* leads to decrease in the length of the feature vector |**x**|; however, small-scale details may be lost for large cells. And a high block size value reduces the ability to suppress local illumination changes. The number of overlapping cells *oc* influences contrast normalization. To ensure adequate normalization, *oc* should be at least half of *cb*. Large overlap values can capture more information, but they produce a larger feature vector **x**. Thus, the length of the feature vector **x** is

$$|\mathbf{x}| = l_H \cdot l_V \cdot cb_H \cdot cb_V \cdot b, \tag{2}$$

where *l* is the number of blocks per image in horizontal (l_H) or vertical (l_V) direction. For both directions, the number of blocks can be expressed as

$$l = \left\lfloor \left(|I| \cdot cs^{-1} - cb\right) \cdot \left(cb - oc\right)^{-1} + 1 \right\rfloor, \tag{3}$$

where |*I*| is size of the object image *I* (in pixels), and $\lfloor \bullet \rfloor$ symbolizes rounding towards minus infinity.

3.2. Search for optimal setting

The simplified version of the grape detector was used as the reliable and efficient platform while developing the solution aimed at detection of grapes in low-resolution images. In order to keep continuity, we used the set T-3 for its training. Considering the relationship between time complexity and performance, we examined possibilities of the detector applications on images resized at 75 %, 50 %, and 25 % of the original size while keeping the aspect ratio 1:1.

The setting of the HOG descriptor is resolution dependent (subsection 3.1); hence, the optimal setting of the HOG descriptor has to be found together with the optimal setting of the classifier for each resolution. Inspired by our previous work [6], we used a one stage grid search algorithm to find the optimal settings. We used two different sets for the training-evaluation process. Within the training phase, we used T-3, while UX (will be introduced in subsection 3.4) was used within the evaluation phase.

Since our goal was to provide a reliable solution with a low time complexity, we searched for a setting giving the best time complexity vs. performance ratio. We formulized the optimization problem as

$$\mathbf{p}^* = \underset{\mathbf{p} \in P}{\arg\max} \left(m'(\mathbf{p}) - |\mathbf{x}'(\mathbf{p})| \right), \tag{4}$$

where **p** is a vector describing a setting of all tuneable parameters, *P* is a set of all considered settings **p**, and **p*** is the optimal setting. The variables *m*' and |**x**'| are normalized performance scores evaluated using the measure *m*, and normalized length of the feature vector **x**, respectively. We used the feature scaling normalization which can be for the *i*-th element of a vector **z** written as

$$z'_i = \left(z_i - \min \mathbf{z}\right) \cdot \left(\max \mathbf{z} - \min \mathbf{z}\right)^{-1}. \tag{5}$$

As the measure *m*, we used acc (1). We considered **p**=(*b*,*cs*,*cb*,*r*,*C*) for the detector with the linear and **p**=(*b*,*cs*,*cb*,*r*,*C*,*σ*) for the detector with the RBF kernel. For both directions, *H* and *V*, the number of overlapping cells was determined by

$$oc = \lceil 0.5 \cdot cb \rceil, \tag{6}$$

where $\lceil \bullet \rceil$ symbolizes rounding towards plus infinity.

For both kernels and all resolutions, we evaluated the detector for $b \in \{9, 10, \ldots, 21\}$, $C \in \{1, 10, 100, 1000\}$, and $r \in \{[0,180], [0,360]\}$. For the RBF kernel, the kernel width $\sigma \in \{1, 10, 100, 1000\}$ was considered for all resolutions. The only resolution dependent parameters were the cell size *cs* and the number of cells in blocks *cb*. We considered square cells and blocks of sizes given by limits specified in table 1.

3.3. Evaluation

In order to keep continuity [6-8], we used all three performance measures (1) for the evaluation of the detector with the optimal settings **p***. The evaluation was performed on images of 100 %, 75 %, 50 %, and 25 % of the original size, while keeping the aspect ratio 1:1. For the evaluation, we used new datasets which will be introduced in subsection 3.4.

Table 1. The smallest (min) and the largest (max) sizes of cells *cs* and blocks *cb* used within the grid search for various sizes of object images (first line), where the size of the images is specified as the percentage of the original size 40 × 40 px.

	100 %	75 %	50 %	25 %
$cs_{\min}$	5 × 5	5 × 5	3 × 3	2 × 2
$cs_{\max}$	10 × 10	8 × 8	6 × 6	3 × 3
$cb_{\min}$	2 × 2	2 × 2	2 × 2	2 × 2
$cb_{\max}$	4 × 4	3 × 3	3 × 3	3 × 3

3.4. Extended datasets

The original datasets consisted of 400 (E and G) or 566 samples (T-3). In order to get informative evaluation results, we formed new extended datasets for the search process (UX) and the evaluation (EX and GX). The sets were based on a new collection of thirty unique vineyard row photos (twenty-five for EX and GX and five for UX). The photos were captured under the conditions specified in [6] at 6 different locations.

The sets comprised of labelled RGB object images of size 40 × 40 px which were created from the photos using an editor [14]. 500 unique 'positive' and 2000 unique 'negative' samples were acquired for each set. The sets were extended using the artificial 'positive' samples (see subsection 2.3), i.e. each new dataset consisted of 2000 'positive' and 2000 'negative' samples. The selection of the samples followed the criteria specified in subsection 2.1. The set UX contained both types of negative samples in the ratio 1:1. Examples of the samples are shown in figure 2 for all types of datasets.

The i-th extended test set of type E was denoted as EX-i where $i \in X$ and $X \in \{1,2,\ldots,5\}$. The same labelling principle was used also for the extended test set of type G. Sets EX and GX with the same index i were based on one pool of five photos and they shared a collection of 'positive' samples.

4. Results evaluation

4.1. Search for optimal setting

Solving the optimization problem (4) according to the specifications given in subsection 3.2, we found the optimal settings $\mathbf{p}^*$ which were summarized in table 2. For each optimal setting, we provided the length of the feature vector (table 3, $|\mathbf{x}|^*$), and the performance which was measured using accuracy (table 4, acc*). For comparison, we presented the smallest ($|\mathbf{x}|_{\min}$) and largest ($|\mathbf{x}|_{\max}$) lengths of the feature vector $\mathbf{x}$ (table 3), as well as, the best (acc$_{\max}$) and the worst (acc$_{\min}$) performance of the detector according to accuracy (table 4).

Table 2. The optimal settings of the detector for various sizes of object images (first line), and for different kernel functions (first column), where *cs* is cell size, *r* is range, *b* is number of bins, *sb* is block size, *oc* is number of overlapping cells, *C* is regularization constant, and σ is kernel width.

	Linear				**RBF**			
	100 %	75 %	50 %	25 %	100 %	75 %	50 %	25 %
b	11	9	11	13	9	9	9	9
cs	9 × 9	7 × 7	6 × 6	3 × 3	9 × 9	7 × 7	6 × 6	3 × 3
cb	2 × 2	3 × 3	3 × 3	3 × 3	4 × 4	3 × 3	3 × 3	3 × 3
r	[0,180]	[0,180]	[0,360]	[0,360]	[0,180]	[0,180]	[0,360]	[0,360]
C	1	1	1	1	1000	100	10	10
σ	–	–	–	–	100	100	10	10

Table 3. Maximal $|\mathbf{x}|_{\max}$, minimal $|\mathbf{x}|_{\min}$, and optimal $|\mathbf{x}|^*$ length of the feature vector $\mathbf{x}$ for various sizes of object images (first line), and for both kernel functions (first column), obtained within the search process performed according to the specification in subsection 3.2.

		100 %	75 %	50 %	25 %
Linear	$|\mathbf{x}|_{\min}$	144	81	81	81
& RBF	$|\mathbf{x}|_{\max}$	6804	3024	3024	1701
Linear	$|\mathbf{x}|^*$	396	324	99	117
RBF	$|\mathbf{x}|^*$	144	324	81	81

Table 4. Best ($\mathrm{acc}_{\max}$), worst ($\mathrm{acc}_{\min}$) and optimal (acc^*) performance of the detector, measured using accuracy, for various sizes of object images (first line), and for both kernel functions (first column), obtained within the search process performed according to the specification in subsection 3.2.

		100 %	75 %	50 %	25 %
	$\mathrm{acc}_{\min}$	0.8318	0.7948	0.8078	0.7460
Linear	$\mathrm{acc}_{\max}$	0.9390	0.9138	0.8913	0.8543
	acc^*	0.9288	0.9073	0.8828	0.8543
	$\mathrm{acc}_{\min}$	0.5000	0.5000	0.5000	0.5000
RBF	$\mathrm{acc}_{\max}$	0.9435	0.9423	0.9085	0.8765
	acc^*	0.9318	0.9245	0.8998	0.8745

4.2. Evaluation

We evaluated performance of the detector, setup according to the specifications summarized in table 2, using accuracy, precession, and recall (1). We summarized the obtained results in table 5 and table 6 for the linear and the RBF kernel, respectively. We provided average values of the measures (bold) for each type of datasets in both tables.

Table 5. Performance of the detector with linear kernel on the extended test sets (first line) for various image sizes (first column), evaluated using accuracy (acc), precision (pr) and recall (re). Average values of the measures are in bold.

		EX-1	EX-2	EX-3	EX-4	EX-5	avg	GX-1	GX-2	GX-3	GX-4	GX-5	avg
	acc	0.9298	0.9218	0.9190	0.9245	0.9168	**0.9224**	0.9365	0.9280	0.9283	0.9295	0.9195	**0.9284**
100%	pr	0.9604	0.9557	0.9505	0.9579	0.9537	**0.9557**	0.9745	0.9688	0.9698	0.9684	0.9595	**0.9682**
	re	0.8965	0.8845	0.8840	0.8880	0.8760	**0.8858**	0.8965	0.8845	0.8840	0.8880	0.8760	**0.8858**
	acc	0.9073	0.8938	0.9043	0.9093	0.8960	**0.9021**	0.9170	0.9040	0.9150	0.9130	0.9020	**0.9102**
75%	pr	0.9483	0.9442	0.9430	0.9585	0.9459	**0.9480**	0.9691	0.9665	0.9658	0.9667	0.9589	**0.9654**
	re	0.8615	0.8370	0.8605	0.8555	0.8400	**0.8509**	0.8615	0.8370	0.8605	0.8555	0.8400	**0.8509**
	acc	0.8878	0.8683	0.8680	0.8820	0.8700	**0.8752**	0.9135	0.8863	0.8865	0.8945	0.8858	**0.8933**
50%	pr	0.9176	0.9216	0.9153	0.9326	0.9209	**0.9216**	0.9715	0.9612	0.9552	0.9598	0.9552	**0.9606**
	re	0.8520	0.8050	0.8110	0.8235	0.8095	**0.8202**	0.8520	0.8050	0.8110	0.8235	0.8095	**0.8202**
	acc	0.8420	0.8285	0.8320	0.8325	0.8348	**0.8340**	0.8825	0.8573	0.8613	0.8590	0.8620	**0.8644**
25%	pr	0.8566	0.8666	0.8597	0.8666	0.8693	**0.8638**	0.9356	0.9261	0.9179	0.9204	0.9249	**0.9250**
	re	0.8215	0.7765	0.7935	0.7860	0.7880	**0.7931**	0.8215	0.7765	0.7935	0.7860	0.7880	**0.7931**

Table 6. Performance of the detector with RBF kernel on the extended test sets (first line) for various image sizes (first column), evaluated using accuracy (acc), precision (pr) and recall (re). Average values of the measures are in bold.

		EX-1	EX-2	EX-3	EX-4	EX-5	avg	GX-1	GX-2	GX-3	GX-4	GX-5	avg
	acc	0.9358	0.9165	0.9228	0.9353	0.9190	**0.9259**	0.9438	0.9243	0.9285	0.9390	0.9235	**0.9318**
100%	pr	0.9589	0.9572	0.9573	0.9658	0.9549	**0.9588**	0.9754	0.9738	0.9693	0.9736	0.9644	**0.9713**
	re	0.9105	0.8720	0.8850	0.9025	0.8795	**0.8899**	0.9105	0.8720	0.8850	0.9025	0.8795	**0.8899**
	acc	0.9273	0.9128	0.9240	0.9263	0.9148	**0.9210**	0.9335	0.9225	0.9325	0.9295	0.9173	**0.9271**
75%	pr	0.9734	0.9640	0.9619	0.9781	0.9689	**0.9693**	0.9871	0.9856	0.9800	0.9853	0.9744	**0.9825**
	re	0.8785	0.8575	0.8830	0.8720	0.8570	**0.8696**	0.8785	0.8575	0.8830	0.8720	0.8570	**0.8696**
	acc	0.9118	0.8910	0.8938	0.9073	0.8898	**0.8987**	0.9225	0.8960	0.9005	0.9088	0.8895	**0.9035**
50%	pr	0.9583	0.9633	0.9555	0.9705	0.9632	**0.9621**	0.9818	0.9748	0.9706	0.9739	0.9626	**0.9727**
	re	0.8610	0.8130	0.8260	0.8400	0.8105	**0.8301**	0.8610	0.8130	0.8260	0.8400	0.8105	**0.8301**
	acc	0.8773	0.8520	0.8610	0.8680	0.8610	**0.8639**	0.8943	0.8660	0.8738	0.8780	0.8650	**0.8754**
25%	pr	0.9289	0.9282	0.9227	0.9391	0.9318	**0.9302**	0.9663	0.9610	0.9511	0.9621	0.9408	**0.9563**
	re	0.8170	0.7630	0.7880	0.7870	0.7790	**0.7868**	0.8170	0.7630	0.7880	0.7870	0.7790	**0.7868**

5. Discussion

We applied the proposed search method for both kernel functions on object images of dimensions 40×40 px, 30×30 px, 20×20 px, and 10×10 px. For both kernels and all considered resolutions, the detector setup according to optimal settings $\mathbf{p}^*$ reached near best possible results according to accuracy (table 4) while keeping the length of the feature vector to the smallest possible values (table 3). For all resolutions, the detector with RBF kernel setting had better time complexity vs. performance ratio (based on comparison of $|\mathbf{x}|^*$ and acc^*).

Together with the resolution, accuracy of the detector progressively decreased for both kernel functions (tables 4-6). A very similar trend was also observed on precision for the linear kernel, and on recall for the RBF one (tables 5-6). The positive correlation was also observed on recall for both kernels, but the trend was almost linear. The only exception was precision of the detector with the RBF kernel (table 6) where small improvement was observed for object images of sizes 30×30 px and 20×20 px.

In summary, the reduction of time complexity via the reduction of the resolution will be always at the expense of the performance. From our point of view, the detector with the RBF kernel function, applied on object images of dimensions 20×20 px (50 % of the original size) is the best choice. With average accuracy about 90 %, average precision about 97 %, and average recall about 83 %, the detector is still competitive with other state of the art solutions. With the length of the feature vector only 81, such implementation of the detector seems to be suitable for real-time applications. For comparison, the HOG descriptor setup according to original specifications (subsubsection 2.2.2) produces vectors of the length 900.

6. Conclusion

The final performance of an image recognition system is determined by many factors. An appropriate setting of all parameters is one of them. The parameters may be introduced in any of the four stages of the vision pipeline. A large number of parameters have to be considered once the HOG descriptor is used. Setting of such a system is not an easy task which was also the case of the grape detector. In order to face this issue, we proposed the method aimed at search for a setting which gives best time complexity vs. performance ratio of image recognition systems based on the HOG features. The application of the method on the grape detector proved the reliability of the method. We therefore expect a wider utilization of this method while developing image recognition systems of similar structure.

Acknowledgements

The work was supported by the Funds of University of Pardubice, Czech Republic (SGS-2017-027). We would like to offer our special thanks to Víno Sýkora s.r.o. company which enabled us to perform experiments in its vineyards.

References

[1] Berenstein R, Shahar O, Shapiro A and Edan Y 2010 Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer *Intell. Serv. Robot.* **3** pp 233-43

[2] Nuske S, Achar S, Bates T, Narasimhan S and Singh S 2011 Yield estimation in vineyards by visual grape detection *Proc. 2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (IEEE) pp 2352-8

[3] Diago M, Correa C, Millán B, Barreiro P, Valero C and Tardaguila S 2012 Grapevine yield and leaf area estimation using supervised classification methodology on RGB images taken under field conditions *Sensors* **12** pp 16988-17006

[4] Liu S and Whitty M 2015 Automatic grape bunch detection in vineyards with an SVM classifier *J. Appl. Log.* **13** pp 643–53

[5] Reis M et al. 2012 Automatic detection of bunches of grapes in natural environment from color images *J. Appl. Log.* **10** pp 285–90

[6] Škrabánek P and Runarsson T 2015 Detection of grapes in natural environment using support vector machine classifier *Proc. of the 21st Int. Conf. on Soft Computing MENDEL 2015* (Brno: Brno University of Technology) pp 143-50

[7] Škrabánek P and Majerík F 2016 Evaluation of performance of grape berry detectors on real-life images *Proc. of the 22nd Int. Conf. on Soft Computing MENDEL 2016* (Brno: Brno University of Technology) pp 217-24

[8] Škrabánek P and Majerík F 2016 Simplified version of white wine grape berries detector based on SVM and HOG features *Artificial Intelligence Perspectives in Intelligent Systems: Proc. of the 5th Computer Science On-line Conf. 2016* Vol 1 (Springer International Publishing) pp 35-45

[9] Dalal N and Triggs B 2005 Histograms of oriented gradients for human detection *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, 2005* Vol 1 (IEEE) pp 886-93

[10] Burges C 1998 A Tutorial on support vector machines for pattern recognition *Data Min. Knowl. Discov.* **2** pp 121-67

[11] Bergstra J and Bengio J 2012 Random search for hyper-parameter optimization *JMLR* **13** pp 281-305

[12] Lampert C 2008 Kernel methods in computer vision *Found. Trends. Comput. Graph. Vis.* **4** pp 193-285

[13] Sokolova M and Lapalme G 2009 A systematic analysis of performance measures for classification tasks, *Inf. Process. Manage.* **45** pp 427-37

[14] Škrabánek P 2015 Editor for marking and labeling of object images for binary supervised classification in MATLAB environment *Proc. of the 21st Int. Conf. on Soft Computing MENDEL 2015* (Brno: Brno University of Technology) pp 151-58