Stream pattern recognition of touching elements of grain mixtures

E A Dobrynin and A A Krylovetsky
Computer Science Department, Voronezh State University, 1 University sq., Voronezh, 394018, Russia
E-mail: aakryl@sc.vsu.ru

Abstract. The paper outlines the problem of stream classification of grain kernels, which involves machine learning methods in cases the images are touching. The applicability of the existing methods of contour separation under real industrial conditions is studied. As a result, a new efficient algorithm is introduced that enables stream separation of grain kernel images. The study has shown that morphological descriptors for restored contours can be used in feature vectors when solving the problem of image classification by means of a neural network with fully connected layers. A program solution is developed that enables classification of the elements of grain mixtures in industrial systems based on the use of 60 the most important morphological, colour and texture descriptors. The paper proposes using the statistical choice of descriptors and the automatic adjustment of the grain mixture flow while performing stream pattern recognition.

1. Introduction
Grain type identification, removal of impurities and poor quality samples are the key stages that precede further grain processing in the modern food processing industry. The range of grain cleaning (or sorting) machines includes mechanical, aerodynamic separators and colour sorters.

A colour sorter enables rejecting hardly separable impurities that are extremely difficult or impossible to remove by means of machines of other classes. It processes images of grain kernels going through a distributing trough that are taken with cameras sensitive to different spectra. In this case, it is not always possible to achieve a separate flow of objects along the distributing trough and to ensure that the distance between their edges is large enough to select them, which hinders sorting. The application of digital image processing methods enables the design of complex and efficient sorting algorithms. The foregoing implies the topicality of this study.

A sorting algorithm intended to be implemented in industrial sorting machines must perform three interconnected tasks:

1. Preliminary binarization of an image taken by one or several cameras of a sorting machine in order to separate the elements it contains from the background.
2. Viewing the selected image elements as a cluster of touching objects of interest (cereal crop kernels and impurities) and performing the separation of these objects by finding their contours in the considered clusters.
3. Classification of the separated objects of interest involving the identification of their belonging to a class of objects from a fixed set (different cereal crops and impurities).

There are several methods for separation of touching grains.
An earlier paper [1] describes an object segmentation algorithm based on the primary morphological operations of erosion and dilation applied to a binary image:

1. Erosion with a gradually increasing structuring elements set (kernel) is applied to a binary image of touching grain kernels (pixels with brightness value 0 denote the background and pixels with other values - objects) several times till all the objects are separated.
2. Erosion separated the grains but distorted them greatly. Therefore, dilation is applied as a next step to increase the objects and fill small holes not letting the kernels touch again.

The method was tested on grain samples of different grades, the results are shown in table 1.

**Table 1.** The results of separating touching grain kernels of different cereal crops based on the erosion-dilation method.

| Crop           | Separation accuracy (%) |
|----------------|-------------------------|
| Hard spring wheat | 95                     |
| Hard wheat      | 95                     |
| Barley          | 94                     |
| Oats            | 79                     |
| Rye             | 89                     |

Achieving sound accuracy in grain separation for wheat and barley, the algorithm shows much poorer performance in cases with oats and rye. The conclusion of the paper [1] emphasizes the low efficiency of this algorithm for long object clusters and clusters that contain long and narrow necks.

There is another approach, which involves the use of morphological watershed segmentation. The concept of watershed is based on presenting an image as a three-dimensional surface with the level of pixel brightness representing its height. In this case, it is possible to identify 3 types of points on the surface:

- local minima points,
- points on the slopes where the water flows down to the center of the basin,
- points on the ridge.

Lines formed by the ridge points are watershed lines, therefore the major objective of this method is the actual search of watershed lines.

The algorithm is described as follows:

1. Holes through which water starts flooding a three-dimensional surface are made at the points of local minima.
2. If the water from the both sides of the ridge is ready to meet and form a common basin, a barrier must be placed there.
3. When there are only barriers left above the water, the algorithm stops. The barriers obtained by means of this method are the required watershed lines.

The algorithm is prone to oversegmentation of images and, like in the case with erosion and dilation, is unable to separate long touching kernels, and clusters with long narrow necks between the objects they consist of.

An interesting approach is described in papers [2] and [3]. The algorithm involves separation based on smoothing the object edges with the elliptical Fourier series and the following analysis of the curvature.

The algorithm is described as follows:

1. The contour of an object is represented in the chain code by means of which size-invariance is achieved while it’s only the description of the form that is kept.
2. The approximation of the original contour with elliptical Fourier series based on the obtained code can be performed using the formulas developed in [4] and [5].
3. The curvature is calculated for each point of the restored contour. According to it we search for junctures using some threshold value. These junctures will be the candidate peaks of the separating lines.
4. A set of potential separating lines is drawn. It equals the Cartesian product of juncture points sets by itself. The set of points pairs will form a list arranged by distance between the junctures.

5. The first pair of points from the list determines the first separating line. Further going down the list implies exclusion of the pairs containing at least one point that belongs to separating lines that have already been determined, which thereby creates new ones.

Figure 1 provides an example of successful separation. It’s clear that the algorithm efficiently handles both the cases of multiple touches and those with clusters containing long narrow necks. However, since the method relies greatly on calculated curvature values and compares them to some threshold one, there can be cases when the curvature in a certain point falls short of the threshold while it is the apparent pair of another point, together forming a separating line.

![Figure 1](image_url)

**Figure 1.** Examples of successful separation of touching grain kernels by smoothing edges method involving elliptical Fourier series and the following analysis of their curvature.

2. **Materials and methods**

The methods described above prove inefficient when separately applied to the analysis of image streams containing grain mixtures elements that are characterized by larger touching areas, sets of simultaneously touching objects and rough contours. Such image features are unavoidable in actual food production, therefore fast and correct processing of such features is an essential property of an algorithm applicable to the operation as a part of a color sorter software.

This paper suggests a new complex algorithm to separate images of grain mixtures elements based on the algorithms described above but showing a better separation quality to required time ratio. Its performance is less affected by the conditions described above. Use of this algorithm enabled the development of a program solution for automated classification of common wheat, red wheat, lentil, red flax and barley kernels.

Basic image analysis methods based on machine learning that are currently employed in machine vision systems to assess the quality of agricultural produce were described in the papers [6-12]. In particular, algorithms extracting information concerning an object’s size, form and colour, algorithms for the calculation of form and colour descriptors, Fourier descriptors and texture descriptors based on Gray Level Co-occurrence Matrices (GLCM) and Gray Level Run-Length Matrices (GLRM) were introduced. The authors developed software implementation to extract 230 properties from colour images of grain objects.

This paper introduces a software solution for automated classification of common wheat, red wheat, lentil, red flax and barley kernels as a result of a carried research and computing experiments with the images of grain mixtures elements provided by OOO “Voronezhselmash”. The solution comprises the following modules:

1. Segmentation module used for the segmentation of objects from the background.
2. Localization module used for localizing separate touching objects in selected clusters of objects.
3. Separate object classification module.

Python was used as the basic programming language. The applied machine vision algorithms are provided by OpenCV library [13]. The artificial neural network of the classification module was implemented using PyTorch [14], whose specific features include a simple program interface and a program capability of building a dynamic network to perform quick experiments with different types of network architecture. PyTorch was used for a range of modern studies [15-18] and is massively supported by Facebook and hardware components manufacturers [19].

The simple debugging user interface of our software was created by means of wxPython library [20] intended for cross-platform graphic interfaces creation.

3. Segmentation of objects from the background
The module for segmentation of objects from the background processes an original image obtained after sorting machine’s work. The following algorithm is used:
1. An RGB image \( I = (I_R, I_G, I_B) \) is converted to monochrome. Experience has shown that accurate segmentation of images with blue backgrounds is possible by simple taking of blue channel \( I = I_B \).
2. At this stage threshold segmentation is applied. The threshold is calculated with Otsu’s method. The result is a binary image with zero intensity background pixels.
3. The OpenCV function findContours() is applied to select contours. It presents the implementation of the algorithms described in the paper [21].
4. The set of contours obtained at the previous stage is filtered for a selected area with boundary values 50 and 3000 established by experiment.
5. The filtered contours are sent to the touching objects localization module.

4. Localization of separate touching objects in selected cluster
The touching objects localization module takes as input objects segmented from the background. The limitations of the separation methods described in the introduction hinder their efficient use as a part of a colour sorter software. Therefore, we introduce a combined approach, which can be represented as the following successive steps at a high level of abstraction:
1. Curvature analysis of the contour, pre-smoothed by elliptical Fourier series, and its following separation using the obtained information.
2. If it is necessary, the further contour separation is performed through successive operations of erosion and dilation.
3. If the steps described above are not effective enough, the separating line is drawn along the short axis of the ellipse inscribed around the studied contour.

It should be noted that the second and the third stages are optional and the appropriateness of their application is determined by the results obtained after the previous step: we must calculate the area of each contour and if it exceeds some threshold value for each grain mixture, the next step of the algorithm is applied.

Testing of the developed algorithm under real industrial conditions proved its more significant efficiency as compared to the methods it was based on. The performance of the algorithm under challenging conditions is shown in figures 2-4.
Figure 2. Cases difficult for separation: (a) is an example of insignificant curvature in the end point of one of the separating lines; (b) is an example of grain kernels large touching area.

5. Classification of a separate object

The software uses an ANN with fully connected layers as a classifier. An ANN with fully connected layers is unable to find image keypoints all by itself and extract the descriptions of the image visual features known as descriptors. Therefore, obtaining a set of features that describes the object on an image as unambiguous as possible is a necessary step preceding the actual model training and use.

Figure 3. Examples of grain kernels large touching areas.

An analysis of the importance of different features is presented in the paper [22]. As a result, 60 the most relevant features were identified. They belong to 3 classes as it is shown in table 2. The listed descriptors are used in the software solution created as a result of this study.

The architecture of the network was designed taking into account the data obtained from several experiments. It is presented in table 3.
Figure 4. Example of multiple touch of grain kernels.

Table 2. Descriptors of grain images.

| Rank | Morphological       | Colour                  | Textural                           |
|------|---------------------|-------------------------|------------------------------------|
| 1    | Mean radius         | Hue mean                | Red GLRM entropy                   |
| 2    | Minor axis length   | Red moment 1            | Green GLRM entropy                 |
| 3    | Area                | Saturation mean         | Grey GLRM entropy                  |
| 4    | Minimum radius      | Blue histogram range 1  | Green GLCM cluster shade           |
| 5    | Perimeter           | Red mean                | Blue GLCM mean                     |
| 6    | Major axis length   | Red histogram range 7   | Red GLCM correlation               |
| 7    | Moment 4            | Red moment 2            | Red GLRM colour non-uniformity     |
| 8    | Maximum radius      | Red histogram range 6   | Green GLRM runpercent              |
| 9    | Moment 3            | Red histogram range 8   | Blue GLRM entropy                  |
| 10   | Boundary Fourier descriptor 20 | Red histogram range 9 | Grey GLRM colour non-uniformity |
| 11   | Boundary Fourier descriptor 2 | Red range            | Red GLCM cluster shade             |
| 12   | Radial Fourier descriptor 2 | Green moment 1        | Blue GLRM colour non-uniformity   |
| 13   | Boundary Fourier descriptor 19 | Red histogram range 10 | Green long run GLRM               |
| 14   | Boundary Fourier descriptor 3 | Blue histogram range 2 | Green run length non-uniformity GLRM |
| 15   | Radial Fourier descriptor 4 | Red range            | Green GLRM colour non-uniformity   |
| 16   | Boundary Fourier descriptor 18 | Green histogram range 1 | Grey run length non-uniformity GLRM |
| 17   | Boundary Fourier descriptor 16 | Blue moment 2        | Blue GLCM entropy                  |
| 18   | Radial Fourier descriptor 19 | Hue range           | Red GLCM entropy                   |
| 19   | Radial Fourier descriptor 5  | Red moment 3          | Red run length non-uniformity GLRM |
| 20   | Radial Fourier descriptor 3  | Green moment 2        | Green GLRM short run               |

Table 3. Architecture of the neural network.

| Input layer size (neurons) | Inner layer size (neurons) | Output layer size (neurons) |
|---------------------------|----------------------------|-----------------------------|
| 60                        | 6                          | 2                           |

Training the classifier model of our software solution involves the following:
1. The input is the function \( c \rightarrow img_s_c \), where \( c \in N, c < K \in N \) is a numeric ID of a class, \( img_s_c \) — an ordered collection of images belonging to class \( c \), \( K \) — the number of classes.
2. The output is a pickle file containing parameters of the mathematical model (ANN) trained to assign objects to one of K classes.

In order to make the further references clearer we will use the following notations:
- \( TrainDS \) is a set of images for model training,
- \( TestDS \) is a set of images for testing the trained model,
- \( |TrainDS| \), \( |TestDS| \) are the sizes of each data set.

The key characteristics of the training procedure are given in Table 4.

| \(|TrainDS| : |TestDS|\) | 70:30 |
|-------------------------------|-------|
| Learning rate                 | 0.001 |
| The number of epochs          | 30    |
| Loss function                 | cross-entropy |
| Optimizer                     | stochastic gradient descent |

Operation of the trained classifier model of our software solution involves the following:

1. The **input** is an image \( img \) and a set of classes of objects identified on \( classes = \{c_i| i = 0..K - 1 \} \), where \( c_i \in \mathbb{N} \) is a numeric ID of a class, \( K \in \mathbb{N} \) — the number of classes.

2. The **output** is the original image \( img \) and conversion \( S = c \rightarrow \) of the numeric IDs of the classes \( c \) into RGB colour values \( (color_R, color_G, color_B) \), that must “colour” pixels forming a \( c \) class object. The image \( img \) is segmented accordingly.

### 6. Conclusion

As a result, we developed a software solution for automated classification of the elements of grain mixtures based on images. Table 5 contains the classification accuracy values for seed elements of grain mixtures that are averages obtained after 10 learning and testing runs.

| Crop            | Classification accuracy (%) |
|-----------------|-----------------------------|
| Common wheat    | 88.52                       |
| Red wheat       | 76.77                       |
| Lentil          | 75.13                       |
| Red flax        | 82.61                       |
| Barley          | 90.43                       |

The testing of the developed software proved the high efficiency of the suggested algorithm for separating the images of touching elements of grain mixtures. Among the factors limiting the recognition accuracy we should name the following:

1. Accurate separation of the touching elements of grain mixtures can be impossible in some cases. As a result, the obtained images of separate grain kernels may contain artifacts hindering the further classification.

2. The size of images of separate grain kernels that we obtain after the separation procedure is usually small, with average of 20×10 pixels. Therefore, images carry a limited amount of information, which complicates the classification.

3. We used a unified average architecture of the neural network and a unified feature set for all cereal crops analyzed in this study.

In order to improve the classification performance, we suggest using cameras of a higher quality, as they enable to get grain kernel images of higher resolution; exploiting the automatic adjustment of speed of the grain mixture flow along the distributing trough to minimize touching and overlapping; completing the algorithm with a statistical choice of descriptors and a neural network architecture adjustment.


Acknowledgments
The results of this study were obtained within the framework of the state task commissioned by the Ministry of Education and Science of Russian Federation within project № 8.384.2017/4.6 “The development of express analysis and classification tools for heterogeneous stream of elements of grain mixtures with pathologies based on the integration of spectral analysis and machine learning methods.”

References
[1] Shatadal P, Jayas D S and Bulley N R 1995 Digital image analysis for software separation and classification of touching grains Transactions of the ASAE 38(2) pp 635–643
[2] Wang W and Paliwal J 2006 Separation and identification of touching kernels and dockage components in digital images Canadian Biosystems Engineering 8 pp 1–7
[3] Mebatsion H K and Paliwal J 2011 Fourier analysis based algorithm to separate touching kernels in digital images Biosystems Engineering 108 66–74
[4] Hiraoka Y and Kuramoto N 2004 Identification of Rhus succedanea L. cultivar using elliptic Fourier descriptors based on fruit shapes Silvae Genetica 53 pp 221–226
[5] Neto J C et al 2006 Plant species identification using Elliptic Fourier leaf shape analysis Computers and Electronics in Agriculture 50 pp 121–134
[6] Majumdar S and Jayas D S 2000 Classification of cereal grains using machine vision. I. Morphology models Transactions of the ASAE 43(6) pp 1669–1675
[7] Majumdar S and Jayas D S 2000 Classification of cereal grains using machine vision. II. Color models Transactions of the ASAE 43(6) pp 1677–1680
[8] Majumdar S and Jayas D S 2000 Classification of cereal grains using machine vision. III. Texture models Transactions of the ASAE 43(6) pp 1681–1687
[9] Majumdar S and Jayas D S 1988 Classification of cereal grains using machine vision. IV. Combined morphology, color, and texture models Transactions of the ASAE 43(6) pp 1689–1694
[10] Karunakaran C, Visen N S, Paliwal J, Zhang G, Jayas D S and White N D G 2001 Machine Vision Systems for Agricultural Products CSAE Paper 01-305 Mansonville QC: CSAE/SCGR.
[11] Paliwal J 2002 Digital image analysis of grain samples for potential use in grain cleaning (Ph.D. thesis) Department of Biosystems Engineering, University of Manitoba, Winnipeg, MB, Canada p 247
[12] Visen N S Machine Vision Based Grain Handling System (Ph.D. thesis) Department of Biosystems Engineering, University of Manitoba, Winnipeg, MB, Canada p 175
[13] Bradski G The OpenCV Library Dr. Dobb's Journal of Software Tools
[14] PyTorch. Available at: https://pytorch.org (accessed 5 October 2018).
[15] Iglovikov V, Seferbekov S, Buslaev A V and Shvets A 2018 TernausNetV2: Fully Convolutional Network for Instance Segmentation (ArXiv e-prints)
[16] Mensch A, Parietal I and Blondel M 2018 Differentiable Dynamic Programming for Structured Prediction and Attention (ArXiv e-prints)
[17] Zeng A, Song S, Welker S, Lee J, Rodriguez A and Funkhouser T 2018 Learning Synergies between Pushing and Grasping with Self-supervised Deep Reinforcement Learning (ArXiv e-prints)
[18] Toyota Research Institute accelerates safe automated driving with deep learning at a global scale on AWS. Available at: https://aws.amazon.com/blogs/machine-learning/toyota-research-institute-accelerates-safe-automated-driving-with-deep-learning-at-a-global-scale-on-aws (accessed 5 October 2018)
[19] Facebook accelerates AI development with new partners and production capabilities for PyTorch 1.0. Available at: https://code.fb.com/ai-research/facebook-accelerates-ai-development-with-new-partners-and-production-capabilities-for-pytorch-1-0 (accessed 5 October 2018)
[20] wxPython. Available at: https://www.wxpython.org (accessed 5 October 2018)
[21] Suzuki S et al 1985 Topological structural analysis of digitized binary images by border following
[22] Paliwal J, Visen N S, Jayas D S and White N D G 2003 Cereal Grain and Dockage Identification using Machine Vision *Biosystems Engineering* 85(1) pp 51–57