Automatic Visible Defect Detection and Classification System Prototype Development for Iron-and-Steel Works

Rustem Alkapov
South Ural State University
Chelyabinsk, Russia

Artem Konyshev
South Ural State University
Chelyabinsk, Russia

Nikita Vetoshkin
South Ural State University
Chelyabinsk, Russia

Nikolay Valkevich
South Ural State University
Chelyabinsk, Russia

Pavel Kostenetskiy
South Ural State University
Chelyabinsk, Russia
kostenetskiypa@susu.ru

***Abstract*—This paper is written as a part of "Automatic visible defect detection and classification system (AVDDCS) development for Electrolytic Tinning Unit" project for Iron and Steel Works. As a part of pre-design research, a system prototype was developed. It consists of Preprocessor, Classifier, Server, Database and User interface. The difference between this prototype and analogs is the use of convolutional neural networks to improve the accuracy of the classification of visible defects.**

Keywords—automatic defect detection, automatic defect classification, artificial neural networks, computer vision

I. Introduction

Nowadays vision-based steel surface inspection systems are produced by many well-known companies (Codnex, EES, Matra, Parsytec, Siemens-VAI, Sipar, etc.). Existing systems detect defects very well, but they still have some problems with defect classification. The reason for these problems is that the steels surfaces often differ in appearance after rolling on different units of the same plant. This leads to the algorithms complication of detecting and classifying defects on the steel surface. In addition, the complexity is caused by many varieties of the surface defects. Characteristics and classification of the defects are not regulated by generally accepted standards and differ for each plant and unit. Even small changes in the production process can cause the appearance of new types of the defects.

The architecture and specifications of commercial steel surface inspection systems generally are not in open access, however there are many research projects on the subject. Caleb and Steuer [1] is an interesting paper where an algorithm based on hand-crafted features in combination with neural network for classification is described. In [2] used the same approach, but for detection task. In [3] a convolution neural network is used for features extraction, and that approach is quite effective.

The process of defects detection on steel surface and their classification in steel surface usually follows three steps [4]:

- Localization of regions of interests (RoIs) by means of segmentation,
- Extraction features from RoIs,
- Classification into defects according to the found features.The AVDDCS is shown in fig. 1. Rectangles denote the modules of the system. The arrows denote the interactions of the modules.

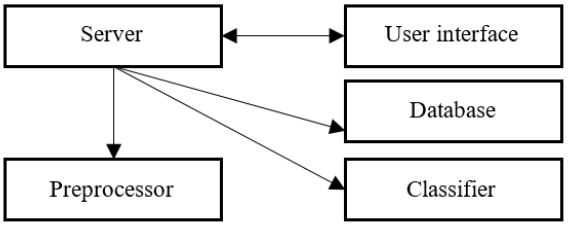

Fig. 1. The AVDDCS scheme

Preprocessor is responsible for image acquisition from cameras, binding images to physical coordinates, splitting images into parts, finding regions of interest (ROI) with visible defects and sending them to classification unit. *Classifier* is responsible for image classification, training data preparation and neural network training. *User interface* is responsible for system management. *Database* stores system data. *Server* is responsible for dataflow control, API provision, business logic.

II. Hardware Selection

A. Computer Vision Test-Bed Prototype

The computer vision test-bed prototype is being developed in Supercomputer Simulation Laboratory of SUSU.

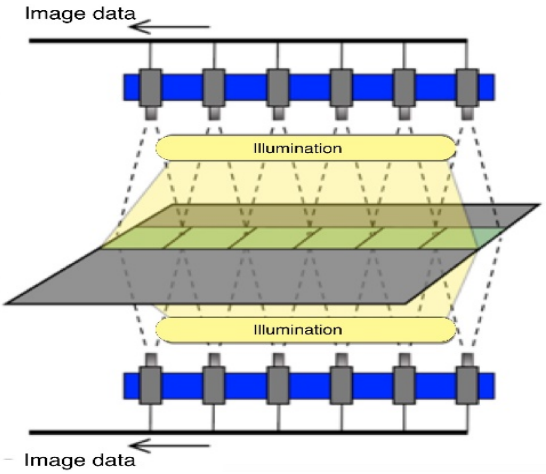

Fig. 2. Test-bed prototype diagram

Test-bed consists of multiple cameras fixed on frame, directed perpendicular to metal sheet and connected to computer. All elements of test-bed are stationary. Test-bed prototype diagram is shown at fig. 2.

B. Camera Interface Selection

Gigabit Ethernet, USB 3.0 Vision and Camera Link interfaces are usually used for computer vision cameras connection. PCI Express bus is usually used to connect correspondent interface controllers. When connecting cameras, the main problem is caused by necessity of having dozens of high-speed interface ports. Expansion cards are used to solve that problem. One PCI Express 3.0 4x expansion card can have either up to 4 USB 3.0 ports that work in parallel and don't share bandwidth or 1 Camera Link port. In case of Gigabit Ethernet cameras, it's enough to use one dual port 10 Gigabit Ethernet network card paired with standard network switch. Camera interfaces comparison is shown in table 1.

TABLE I. CAMERA CONNECTION INTERFACES

Interface	Max band-width (Gbit/s)	Max cable length (m)	Expansion card price per 1 cam-era ($)	Active repeater for cable exten-sion price ($)
Gigabit Ethernet	1	100	38	Not required
USB 3.0 Vision	4.8	3	20	200
Camera Link Extended Full	6.8	3	330	650

C. Required Camera Parameters Calculation

The developed prototype is focused on the use of area scan cameras, as, according to the authors' observations, commercial steel surface inspection systems of high resolution, working with very small defects (0.5 mm square) and high speed of the strip (7 m/s) are built on their basis. Line scan cameras are usually used only as an auxiliary part of such systems to search for specific types of defects.

Input parameters for camera amount calculation and maximum sheet speed calculation:

- minimum visible defect width and height (mm) – *min_defect_size*;
- minimum pixel amount in width and height required for confident detection of smallest defect – min_defect_pix-els;
- metal sheet parameters – *width* (mm); *speed* (m/s);
- percent of overlapping of images from neighboring cam-eras – overlap;
- camera parameters – matrix width *x* and height *y (pixels)*.

The maximum physical dimensions of the rectangle (X*Y mm) on a metal sheet, which can be recorded by one camera with a resolution enough to detect the minimum defect, is calculated by formulas:

$$X = \frac{x * min\_defect\_size}{min\_defect\_pixels}$$

$$Y = \frac{y * min\_defect\_size}{min\_defect\_pixels}$$

The minimum frame rate of the camera at a known speed of the metal sheet is calculated by the formula:

$$fps = \frac{speed * 1000}{Y}.$$

The number of cameras that is needed to physically recognize the defect on both sides of a metal sheet is calculated by the formula:

$$n = 2\left\lceil \frac{width}{X * \left(1 - \frac{overlap}{100}\right)} \right\rceil$$

TABLE II. GIVEN VALUES

Name	Value
min_defect_size	0.5 mm
min_defect_pixel	3
width	1650 mm
speed	7 m/s
overlap	5%

For example, Basler® monochrome infrared cameras listed in table 3 were reviewed. Monochrome cameras were used to reduce the amount of data transferred from cameras, and to speed up preprocessing algorithms, that usually use monochrome images. The average cost of lens, which is needed for each camera is $600.

TABLE III. CAMERAS CHARACTERISTICS

Model	In-ter-face	Reso-lu-tion (px)	FPS	Max sheet spee d (m/s)	Cam era price , $	Pric e with ac-ces-so-ries	Q ua nti ty	Sum , $
acA1300 60gm NIR	GE	1280x 1024	60	10,2	754	1392	18	2505 6
acA2000 50gm NIR	GE	2048x 1088	50	9,07	1699	2337	12	2804 4
acA2000 165um NIR	USB 3	2048x 1088	165	29,9	1526	2326	12	2791 2
acA2040 90um NIR	USB 3	2048x 2048	90	30,7	1806	2606	12	3127 2
acA2040 180km NIR	Cam era Link	2048x 2048	180	61,4	2506	3756	12	4507 2

III. IMAGE PREPROCESSING UNIT

Preprocessor receives a stream of images from cameras with a certain frequency and divides them into parts of the equal size. Then, each part is checked for defects by computer vision algorithms. The image parts where the defect is present are sent to the defect classification module. Also, for debugging

purposes, the module supports stitching multiple images into one and highlighting defects on the image.

The preprocessing unit is controlled by the operator, which is a user interacting with the system. Preprocessing unit's output is transferred to the classification unit, which is a software unit that receives information from the preprocessing unit. The operator can view the video stream by selecting its type - a stitched video stream or video stream from a specific video camera. In the first case, images from all cameras are stitched into one. In the second case, the images from the cameras is shown without stitching. The operator can also calibrate the cameras, i.e. change image stitching parameters. Classification unit can receive a region of interest, i.e. get an image with a defect, obtained during the work of the preprocessing unit.

A. *Image Stitching and Blending*

Due to the large width of the metal sheet, one camera cannot completely capture its surface, while retaining enough resolution to detect defects, so the prototype uses a lot of cameras located in parallel. Images from all cameras must be stitched together into one panoramic image for more convenient monitoring of the process of detecting defects by the operator and for obtaining a complete image of a defect captured by two or more cameras.

Usually, panorama creating algorithm includes the step of finding features and comparing them to features of another image. This step can be skipped during the defect detection, since it is difficult to find features on the metal sheet that are not defects, and the displacement of the cameras relative to each other are fixed and known in advance.

The next step in the construction of a panoramic image is the stitching of images and the brightness alignment. To stitch the images, the gradient blend algorithm [5] was chosen, because of the simplicity of the implementation and less visible seam than in the alpha blend algorithm.

B. *Defect Detection*

It is necessary to detect areas of the image, which are highlighted from the general background, since they can be defects.

The input data of detection algorithm is grayscale image. Then the Sobel operator [6] is used to find gradients along two axes. After this, two binary images are constructed by thresholding [7]. At this step, white areas on binary images show areas of grayscale image that vary along two axes. To improve the results, the morphological opening [8] is performed and both images are superimposed on each other. Then a morphological closure is made, and the contours of the white areas are located, which are then filtered by the area relative to the minimum defect area.

C. *Designing of the Module*

Because of the requirements analysis, the following functional requirements were identified. The system must:

- detect visible defects on the surface of the metal sheet;
- provide an API for obtaining images of detected defects;
- stitch multiple images into one;
- receive images from multiple cameras at a certain frequency;
- be able to display an image in the debug interface;
- provide a user interface for setting image stitching parameters;
- divide the original images into parts of the equal size;
- be able to highlight identify visible defects on the image.

D. *Development of the Module*

The following classes were created for data storage:

- ImagePart – an entity that stores part of the original image and its metadata;
- Config – an entity that stores values from the configuration file.

The following behavior classes were created for data management:

- Preprocessor – main class that initializes the rest of preprocessor unit's classes;
- Splitter – class that splits images into parts of equal size;
- Detector – class that detects defects;
- Blender – class that stitches multiple images into one.

The following classes were created for interaction with external systems:

- Streamer – class that connects preprocessor unit to cameras and obtains images;
- Sender – class that sends parts of images with metadata to classification unit.

Main preprocessing cycle activity diagram is shown at fig. 3.

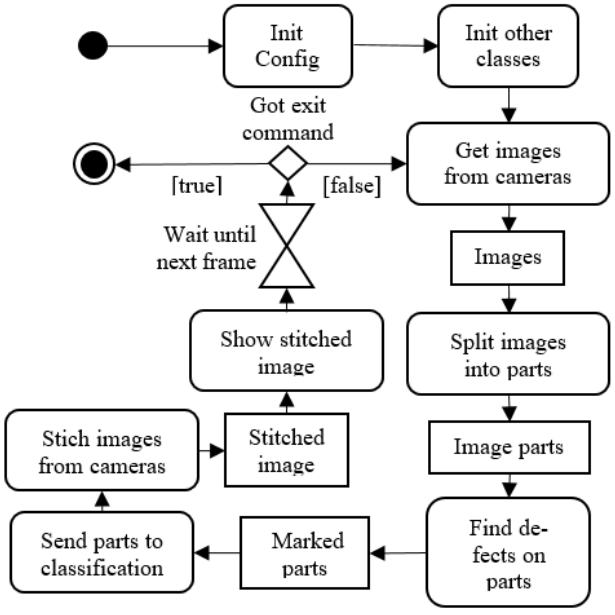

Fig. 3. Main preprocessing cycle activity diagram

The original image is split into parts of the equal size. The size of the parts is determined by the input format of the neural network in the classification unit.

Defect detection algorithm activity diagram is shown at fig. 4.

The main idea of the defect detection algorithm is that visible defects have boundaries, at which the pixels brightness in gray-scale varies sharply. Having found these boundaries with the help of the Sobel operator on two axes, it is possible to single out the defect completely.

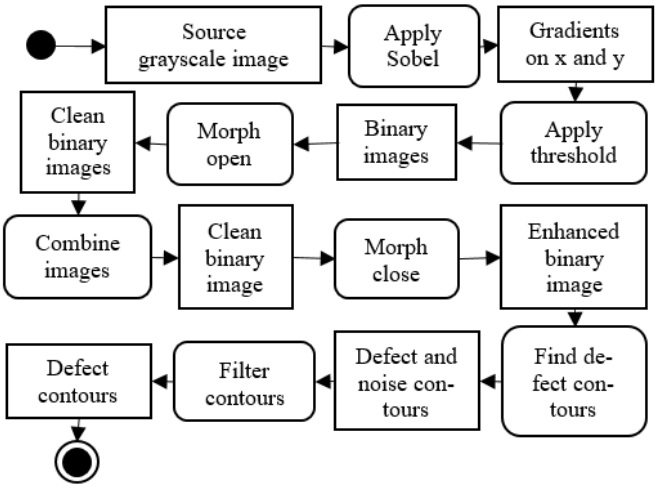

Fig. 4. Defect detection algorithm activity diagram

If the resulting contour array is not empty, then the part of the image on which the defects were searched is marked with the has_defects flag and is subsequently sent to the classification unit. Also, the resulting contour array can be used to highlight defect in the image (fig. 5).

E. Computational Experiments

Computational experiments were carried out on the images of surface defects of hot rolled metal from Northeastern University surface defect database [9]. Because of the experiments, the numerical coefficients of the defect detection algorithm were chosen, which make it possible to detect the greatest number of defects.

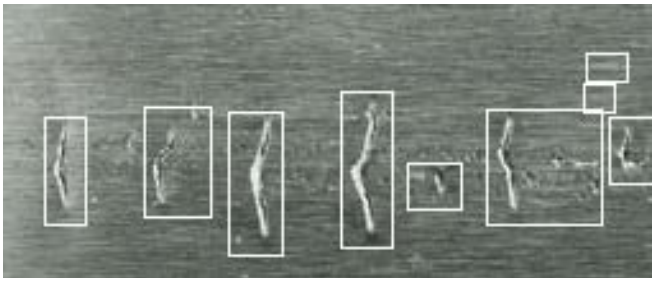

Fig. 5. Example of detecting and highlighting defects

IV. Image Classification Module Based On Artificial Neural Networks

Classifier is responsible for image classification, training data preparation and neural network training.

A. Classification methods

The most commonly used classification methods for visible defects of steel surface are neural network and support vector machine. Other methods for classification are k-nearest neighbors, fuzzy logic, discriminant function-based methods, self-organizing map and learning vector quantization [4].

Neural networks. Many different types of defects can be observed on steel surface. That's why neural network broadly used [1], [10]. Neural network commonly used with one hidden layer. Large variation of number of nodes used in input layer (feature space: 4 to more than 50), hidden layer (10 to 100) and output layer (defect class: 2 to 24). Most of the applications of neural networks occur in steel surface classification of both hot and cold metal rolling, due to the manifestation of a large number of important classes. Comparison of the performance of neural networks with other classifiers shows that neural network are the most productive in this field [1], [10].

Support Vector Machine (SVM). Initially, this method was used to separate two classes, and now it's used for classification of two fairly similar classes [11], [12]. Later, the method was extended to separate defects into multiple classes using the following approaches: «one-vs-all» and «one-vs-one» [13], [14]. To apply multiclass classification a few binary classifiers are required to be trained. The most common approach is the «one-vs-all», where a classifier is trained as positive label for one class and negative label for all other classes. This strategy requires classifier for each class. The voting strategy following a series of binary classifiers described in [15], but SVM requires more resources than neural network and some other classifiers.

Due to the high efficiency of neural networks, both in terms of accuracy and performance, that approach is chosen to solve this problem.

The image of defects from the Northeastern University surface defect database [9] was used to develop the module and train the neural network at the stage of pre project studies. The collection consists of six classes of typical defects that appear on the surface of metals during hot rolled sheets. Images are presented in black and white, with a size of 200x200 pixels with 300 pieces per class. Among defects the following classes are listed: rolled-in scale, patches, crazing, pitted surface, inclusion and scratches.

B. Functional requirements

- The module should provide the ability to classify the defect of rolled metal in the incoming image with an accuracy of at least 95%.
- The module should provide the ability to add new class of defects to the ANN training dataset.
- The module should provide the ability to train classifier using new samples.

The classification module provides following interfaces: 1) using the classification of defects on the input image; 2) data preparation for model training; 3) model training.

C. Neural network architecture

The architecture of the neural network was built in according to the principles laid down in the implementation of LeNet 5 [16], except activation function: sigmoid replaced with rectified linear unit (ReLU). The main advantages of choosing ReLU are listed below.

- The calculation sigmoid requires the execution of resource-intensive operations such as exponentiation. ReLU can be realized by threshold transformation of activation matrix at zero.
- ReLU is not saturated. When saturating the sigmoid from one side or the other one (0, 1), the gradient in these areas becomes close to zero, hence the total gradient is reset, which leads to poor learning ability of the network.
- ReLU significantly increases the rate of convergence of the stochastic gradient descent in comparison with the sigmoid [17].

The development of neural network architecture has passed iteration actions of comparisons of trained models' accuracy with different sets of parameters. The architecture of ANN for six classes is shown on fig. 6.

The artificial neural network shown on fig. 6 consists of several blocks described below. White blocks represent the input and output layers. Orange blocks represent convolution layers, green blocks are max pooling layers. Blue blocks represent flatten layer. Red blocks represent dense layers. Dark yellow represent dropout layer. The numerical parameter near the name of the layer is the size of their output tensor after applying this layer. As for dropout layer, the numerical parameter is part of dropped out neurons.

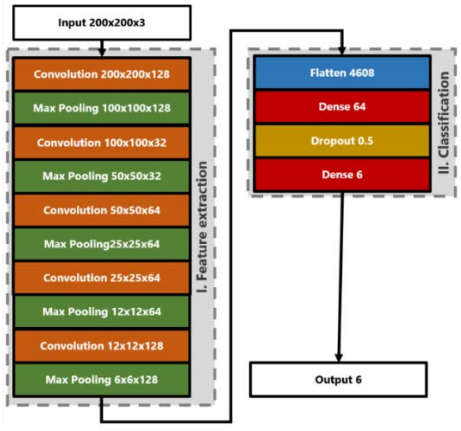

Fig. 6. The architecture of ANN

In addition, the neural network can be divided into two regions: region I represents feature extraction model, which converts input dimension into 128 feature maps of 6x6 size. Region II is responsible for defects classification. The neural network consists of 465126 parameters 295366 of which occur in defects classification region.

D. Data preparation

The data set contains 1800 marked images. In this case it's important to augment data to prevent overfitting. For this purposes ImageDataGenerator class from Keras library is used. Abstract class DataPreparer is the foundation for future data preparation. The NeuDataPreparer class implements DataPreparer and allows to prepare data from defect database from the Northesern University (Boston, USA).

E. Programming implementation of the neural network

The Sequential model is a linear stack of layers which is the base for programming implementation of described neural network. It's possible to attach components in order to make neural network. Required components are described below.

To prevent overfitting the dropout layer is used, usually using this layer improves the learning efficiency and quality of the result [18]. The output layer represents probability vector of class membership as result of Softmax function [19].

The Adam optimizer is chosen as optimization function [20], its parameters are selected during computational experiments.

As a loss function the categorical cross entropy is chosen, this function is used in Keras for multiclass classification.

The important part of neural network training is the creation of callback functions, which allows to observe the model at each epoch of training. These functions are passed as neural network's parameters:

- check_point – this function allows to save the best model (quality depends upon metrics which also is input parameter of neural network, in this case metric is accuracy on validation set) during the training process;
- early_stopping – this function allows to stop training process if accuracy has not improved within several epochs;
- reduce_lr – this function allows to decompose learning rate; if accuracy has not improved within several epochs, then the learning rate will be multiplied by the expansion factor in range from 0 to 1.

The final criterion of the training stage is the global parameters. The number of steps of each epoch (steps_per_epoch) is the total number of steps performed by the generator of the training set in order to finish current epoch. Same for validation set (validation_steps). The input_shape parameter used to specify the size of input layer of neural network. The meaning of the following parameters is obvious from their name: base_learning_rate, learning_rate_factor, min_learning_rate, learning_rate_patience. The number of total epochs equals to 1000, but in practice the number if epoch depends upon callback function (early_stopping).

F. Model implementation

To implement model the Keras library with the Tensorflow backend are used.

In order to convenience the SteelDefectClassifier class is developed. This class provides a simple way to train model and classify an input image.

The __normalize method normalizes input image in the same way as the transformations that are used during the training process – z-score [21]. The result of transformation is shown on

fig. 7. Normalization brings the brightness and brightness spread on the input images into one range.

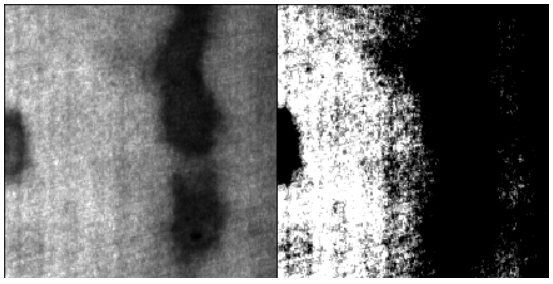

Fig. 7. Defect's illustration before and after normalization

The __get_callbacks method returns list of callback functions, the __get_model method combines Sequential model which was described above.

The public methods provide the described features. The classify method provides the ability to classify an input set of images. The input is list of images (tensors), each has the following spatial characteristics – 200x200x3 (width, height, number of channels). The result of the method is a list of defect classes, the number of which corresponds to the number of received images. In order for the classification to be done it is required to load the model of the neural network. The load_model method allows to load model on specific path. The model which was trained on Northeastern University database is located in the same folder as source code. It's possible to support several models on runtime. The fit_model method is responsible for model training on specific DataPreparer with specific batch size which is passed as an argument.

The prepare_data method allows to prepare source data set for future model training. It follows with the fit_model method of the classifier which allows to train model.

Implementation, training and testing of models were performed on NVIDIA GeForce GTX 1080 Ti GPU, with 3584 NVIDIA CUDA Cores, 11 Gb RAM and frequency 1594 MHz.

The model which was trained with NeuDataPreparer shows accuracy of 98.16% on the test data (with an error of up to 0.5%). Average time of classification per image is 2.1 milliseconds. This result is consistent with other works shown in table 4.

TABLE IV. COMPARISON OF RESULTS WITH OTHER WORKS

Model	Input shape	Execution time	Accuracy	The number of defects
Current model	200x200x3	2.1 ms	98,1%	6
Yi. L., Li. G. and Jiang M. [22]	114x114	1.3 ms	99%	7
Wu. G. [2]	-	40 ms	88%	4
Masci J. [23]	150x150	6.2 ms	90%	7

V. GRAPHICAL USER INTERFACE OF OPERATOR'S AUTOMATED WORKPLACE

The graphical user interface (GUI) was developed for the interaction of the operator and AVDDCS. The GUI allows operator to get real-time statistics about the number of defects, their type and significance. Based on the received information, the operator can make decisions about stopping the rolling mill or reduction of the metal grade. All detected defects are documented in the current steel coil control protocol.

A. The Functional Requirements for the GUI

The operator's automated workplace interface must provide the following main features:

- promptly notify the operator of a new defect on the rolled steel coil;
- display image and information of detected defect;
- display and print out the defects statistics;
- display list of produced steel coils;
- form and print out steel coil control protocols and defects reports;
- display the current state of the steel coil;
- display diagnostic information about the health of the hardware imaging system;
- display diagnostic information about the health of other components of AVDDCS;
- provide users management in the system;
- edit information about steel coils and defects.

B. Implementation of the GUI

The graphical interface was implemented in the programming language C# in the Visual Studio 2017 programming environment using the Microsoft .NET Framework software platform.

The main window of the graphical interface of the operator's automated workplace is shown in fig. 8. The window displays information about the last detected defect, the steel coil card, the data of produced steel coils, diagnostic information about the system state and information about the production process. Steel coils data contain id, material, thickness, start and end time of production. The steel coil card contains information about detected defects on the top and bottom sides of the metal. Information about the last detected defect includes a defect image and a brief description. The defect description consists of id, class, criticalness, size and location, detection time, the number of the video camera that detected defect and the side of the steel coil where defect is located.

The operator can enable or disable automatic updating of information on the screen about the last detected defect. Also, he can enable or disable the sound notification about the appearance of a defect.

At the top of the main window there is information about the state of the cameras health. Each camera has its own indicator. The same way, information about the remaining components of AVDDCS is displayed.

Information about the production process (current steel coil length and rolling speed) is in the upper right corner of the window.

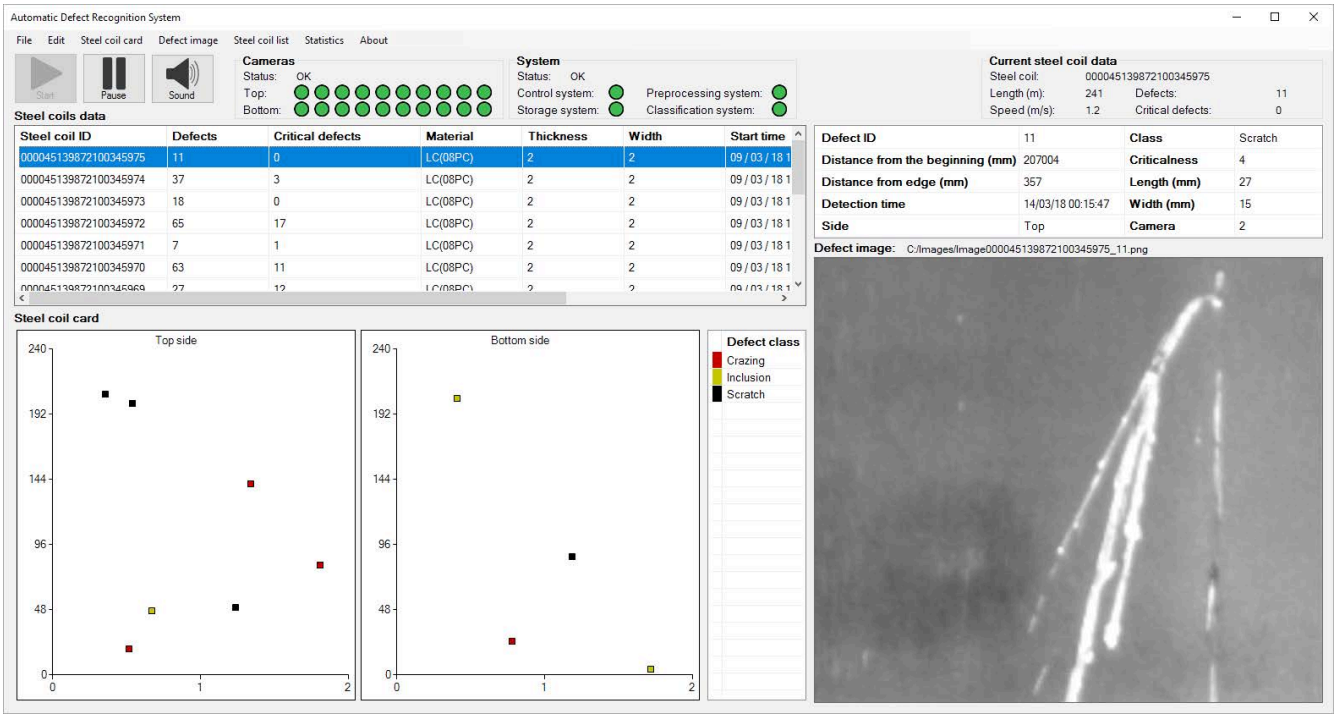

Fig. 8. The main window GUI of operator's workstation

The coil list window allows you to select a coil and move to the steel coil control protocol. In addition, the window allows you to search for coils that meet the specified criteria in the database.

The coil control protocol window contains production information about the coil: ID, material, thickness, start and end time of the coil production, information about the defects classes on the coil, the number of each class defects and the percentage of the area occupied by defects. The protocol contains a coil card of the top and bottom sides, where the detected defects are displayed. Also, there are buttons for editing and printing protocol in the coil control protocol window.

The defect report window contains a picture and description of the defect, its class and physical location on the coil. The defect report window also contains buttons for printing and editing the defect.

The defect editor window allows you to edit information about an existing defect or add a new defect. This is necessary if the system incorrectly classified or missed the defect.

To interact with other components of AVDDCS, the graphical interface of the operator's automated workplace has an API. The interaction is based on the REST architecture. The data is transmitted in JSON format. Web sockets are used to display the status of the unit and AVDDCS in real time.

ACKNOWLEDGMENT

This work was supported in part by Act 211 Government of the Russian Federation, contract № 02.A03.21.0011 and by the Ministry of education and science of Russian Federation, government order № 8.9694.2017.

The reported study utilized the supercomputer resources of South Ural State University [24].

VI. CONCLUSION

In this paper, we proposed the Automatic visible defect detection and classification system prototype for the metallurgical plant, consisting of preprocessing, classification, server, database and user interface units. To improve the classification results, the training samples are collected and augmented for training an artificial neural network. Preliminary experiments show a sufficiently high accuracy of defect classification, due to the use of convolutional neural networks.

REFERENCES

[1] P. Caleb and M. Steuer, "Classification of surface defects on hot rolled steel using adaptive learning methods," KES'2000, Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. Proceedings (Cat. No.00TH8516), vol. 1, pp. 103–108, 2000.

[2] G. Wu, H. Kwak, S. Jang, K. Xu, and J. Xu, "Design of online surface inspection system of hot rolled strips," 2008 IEEE International Conference on Automation and Logistics, pp. 2291–2295, 2008.

[3] M. Popat and S. V. Barai, "Defect detection and classification using machine learning classifier".

[4] N. Neogi, D. K. Mohanta, and P. K. Dutta, "Review of vision-based steel surface inspection systems," EURASIP J. Image Video Process., vol. 2014, no. 1, 2014.

[5] V. Rankov, R. J. Locke, R. J. Edens, P. R. Barber, and B. Vojnovic, "An algorithm for image stitching and blending," SPIE Newsroom, vol. 5701, no. March, pp. 190–199, 2005.

[6] S. Gupta and S. G. Mazumdar, "Sobel Edge Detection Algorithm," Int. J. Comput. Sci. Manag. Res., vol. 2, no. 2, pp. 1578–1583, 2013.

[7] N. Senthilkumaran and S. Vaithegi, "Image Segmentation By Using Thresholding Techniques For Medical Images," Comput. Sci. Eng. An Int. J., vol. 6, no. 1, pp. 1–13, 2016.

[8] D. Chudasama, “Image Segmentation using Morphological Operations,” Int. J. Comput. Appl., vol. 117, no. 18, pp. 16–19, 2015.

[9] K. Song and Y. Yunhui, “NEU surface defect database”.

[10] M.R. Yazdchi, A.G. Mahyari, and A. Nazeri, “Detection and Classification of Surface Defects of Cold Rolling Mill Steel Using Morphology and Neural Network,” 2008 International Conference on Computational Intelligence for Modelling Control & Automation, pp. 1071–1076, 2008.

[11] C. Park and S. Won, “An automated web surface inspection for hot wire rod using undecimated wavelet transform and support vector machine,” 2009 35th Annual Conference of IEEE Industrial Electronics, pp. 2411–2415, 2009.

[12] Hongbin Jia, Y.L. Murphey, Jinajun Shi, and Tzyy-Shuh Chang, “An intelligent real-time vision system for surface defect detection,” Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, vol. 3, pp. 239–242, 2004.

[13] S. Ghorai, A. Mukherjee, M. Gangadaran, and P. K. Dutta, “Automatic Defect Detection on Hot-Rolled Flat Steel Products,” IEEE Trans. Instrum. Meas., vol. 62, no. 3, pp. 612–621, 2013.

[14] Jie Zhao, Yongmin Yang, and Ge Li, “The cold rolling strip surface defect on-line inspection system based on machine vision,” 2010 Second Pacific-Asia Conference on Circuits, Communications and System, pp. 402–405, 2010.

[15] S. Batsuuri, J. Ahn, and J. Ko, “Steel surface defects detection and classification using SIFT and voting strategy,” Int. J. Softw. Eng. Its Appl., vol. 6, 2012.

[16] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, 2017.

[18] G. Hinton, “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups,” IEEE Signal Process. Mag., vol. 29, no. 6, pp. 82–97, 2012.

[19] J. S. Bridle, “Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition,” in Neurocomputing. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 227–236.

[20] S. Ruder, “An overview of gradient descent optimization algorithms,” Biometrics, vol. 55, no. 2, pp. 591–596, 2016.

[21] E. Kreyszig, Advanced Engineering Mathematics. John Wiley & Sons, Inc., 2011.

[22] L. Yi, G. Li, and M. Jiang, “An End-to-End Steel Strip Surface Defects Recognition System Based on Convolutional Neural Networks,” steel Res. Int., vol. 88, no. 2, pp. 176–187, 2017.

[23] J. Masci, U. Meier, D. Ciresan, J. Schmidhuber, and G. Fricout, “Steel defect classification with Max-Pooling Convolutional Neural Networks,” The 2012 International Joint Conference on Neural Networks (IJCNN), pp. 1–6, 2012.

[24] P. S. Kostenetskiy and A. Y. Safonof, “SUSU Supercomputer Resources,” Proceedings of the 10th Annual International Scientific Conference on Parallel Computing Technologies (PCT 2016), pp. 561–563, 2016.