Machine-learning-based Surface Defect Detection and Categorisation in High-Precision Foundry

Iker Pastor-López, Igor Santos, Aitor Santamaría-Ibirika,
Mikel Salazar, Jorge de-la-Peña-Sordo and Pablo G. Bringas
S^{3}Lab, DeustoTech - Computing, University of Deusto, Bilbao, Spain
Email: {iker.pastor, isantos, a.santamaria, mikel.salazar, jorge.delapenya, pablo.garcia.bringas}@deusto.es

***Abstract*—Foundry is an important industry that supplies key castings to other industries where they are critical. Hence, foundry castings are subject to very strict safety controls to assure the quality of the manufactured castings. One of the type of flaws that may appear in the castings are defects on the surface; in particular, our work focuses in inclusions, cold laps and misruns. We propose a new approach that detects imperfections on the surface using a segmentation method that marks the regions of the casting that may be affected by some of these defects and, then, applies machine-learning techniques to classify the regions in correct or in the different types of faults. We show that this method obtains high precision rates.**

I. Introduction

Foundry is one of most ancient processes of the human history and it has been used as a indicative of the progress of a society. Basically, it consists in melting a material and pouring it into a mould where it solidifies into the desired shape. The resulting castings are supplied to key sectors such as aeronautic, automotive, weaponry or naval industries. As one may think, when it comes to these industries the tiniest defect may become fatal. Therefore, foundry is subject to very strict safety controls to ensure the final quality of the manufactured castings.

There are several defects that may appear in the manufactured castings. In this work, we focus on the so-called surface defects. In particular, we focus on inclusions, which are little perforations caused by an excess of sand in the mould; cold laps, which are produced when part of the melted material is cooled down before the melting is completed; and misruns that appear when not enough material is poured into the mould.

Currently, the visual inspection and quality assessment is performed by human operators [1]. Albeit people can perform some tasks better than machines, they are slower and get easily exhausted. Besides, operators are hard to find and to maintain in the industry since they require capabilities and learning skills that usually take them long to acquire. There are also cases of boredom that may affect the process. In some applications, the inspection is critical and dangerous and computer vision can replace more efficiently and without danger [2].

Computer vision has become an important field that can aid to the visual inspection in quality control processes. Computer vision systems are replacing manual inspection in many industries such as timber [3], textile [4] or metallurgical [5], [6] businesses. Whereas manual inspection strongly depends on human factor, computer vision is independent, with the subsequent avoidance of errors.

Against this background, we propose a new approach capable of detecting and categorising inclusions, cold laps and misruns. First, we deploy a machine vision system that retrieves the information about the surface of the tested castings. Second, a segmentation method, based on modelling the correct castings, is used in order to detect the possible defects. Finally, we employ several features extracted from the machine-vision and segmentation systems to train machine-learning algorithms to categorise the possible defects.

Summarising, our main contributions are: (i)an adaptation of a machine vision system to the segmentation of foundry casting regions, (ii) a machine-learning approach to categorise faulty regions on the foundry castings and (iii)an empirical validation using actual foundry castings of our proposed approach.

II. Foundry Processes and Surface Defects

Although all of the foundry processes are not the same, the work-flow performed in foundries is very similar to the work-flow shown in Fig. 1. The most important stages are the following [7]:

- **Pattern making:** In this step, moulds (exteriors) or cores (interiors) are produced in wood, metal or resin and are used to create the sand moulds in which the castings will be made.
- **Sand mould and core making:** The sand mould is the most widely extended method for ferrous castings. Sand is mixed with clay and water or other chemical binders. Next, the specialised devices create the two halves of the mould and join them together to provide a container in which the metals will be poured into.
- **Metal melting:** In this stage (see 1 in Fig. 1), raw materials are melt and mixed. Molten metal is prepared in a furnace and depending on the choice of the furnace, the quality, the quantity and the throughput of the melt change.
- **Casting and separation:** Once the mixture is made, the molten material is poured into the sand mould. It can be done using various types of ladles or, in high volume foundries, automated pouring furnaces. Then, the metal begins to cool down. This step (see 2 in Fig. 1) is really important because the majority of the defects can

appear during this phase. Finally, when the casting has been cooled enough to maintain the shape, the casting is separated from the sand. The removed sand is recovered for further processing.

- **Removal of runners and risers:** Some parts of the casting that had been used to help in the previous processes are then removed. They can be detached through knocking off, sawing or cutting.
- **Finishing:** To obtain a valid result additional actions are usually required, e.g., cleaning the residual sand, heat treatment and rectification of defects by welding.

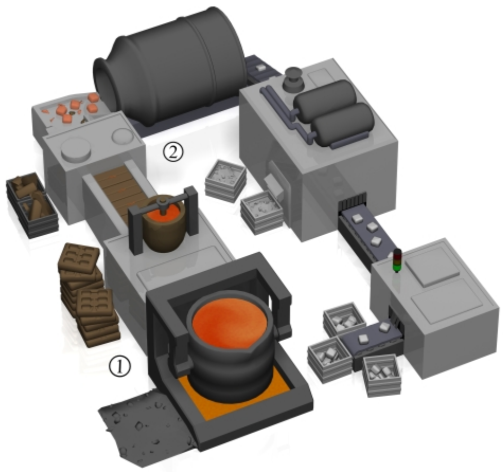

Fig. 1. Foundry process work-flow showing the different stages castings have to pass through. More accurately, in 1 it is performed the metal melting step, and in 2 it is performed the casting preparation and separation step.

In these phases several defects may appear on the surface of the castings. In this work we focus on cold laps, inclusions, and misruns. Cold laps may arise when the pouring phase of the process is performed in more than one step. This implies that regions of the casting begin to cool down whilst material is still being poured into the mould. Inclusions are defects produced by an excess of sand in a region of the casting. Misruns emerge by a lack of material in the casting, generally as a result of not enough material enters into the mould, resulting in a poor filling of the mould.

III. Data Acquisition and Segmentation through Machine Vision

In order to retrieve the data and to process the information, we develop a simple computer-vision system that is composed of a laser camera with 3D technology, a computer with high data processing capabilities and a robotic arm, similar to the one proposed in [8]:

In our case, we can divide this system in the following components:

1) **Image device:** We obtain the three-dimensional data through a laser-based triangulation camera. By taking advantage of the high-power (3-B class) laser, we are able to scan the casting even though their surface tends to be dark.
2) **Processing device:** We utilise a high-speed workstation. In particular, we have used a workstation with a XENON E5506 processor working with 6GB of RAM memory and a QUADRO FX1800 graphic processing unit. This component controls the camera and the robotic arm. Besides, it processes the information retrieved by the image capturing device and transforms it to segments.
3) **Robotic arm:** The function of the robot is to automate the gathering phase of the system, making every necessary move to successfully acquire the data. There are two working options [9]: (i) to employ the arm in order to handle the tested castings, leaving the image device in a fixed position or (ii) to attach the camera to the robotic arm. We selected the second one due to the diversity of the castings.

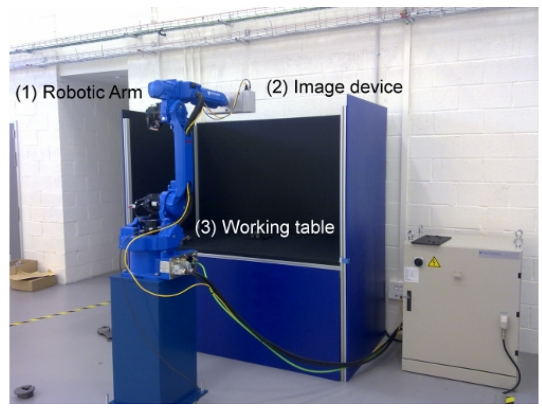

Fig. 2. The architecture of the machine vision system. (1) is the robotic arm of the system, (2) is the image device of the system and (3) is the working table where the castings are put for analysis.

The casting is put on a working table using a manually adjusted foundry mould. The mould is built with a material similar to common silicone, which is easily malleable. In the case we decide to change the casting type, we would only have to change the mould. In this way, we ensure that the vision system allows us to analyse every type of casting in the same position.

Using this architecture, we gather the information of the castings. To this end, we put the casting on the mould and we start the surface scan. The robotic arm makes a lineal movement, retrieving a set of profiles based on the generated triangulation of the laser and the optical sensor. In other words, a foundry casting $\mathcal{C}$ is composed of profiles $\mathcal{P}$ such as $\mathcal{C} = \{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_{n-1}, \mathcal{P}_n\}$. Each profile is retrieved with a thickness of 0.2mm. These profiles are vectors $\vec{p}$ composed of the heights of each point $p_{x,y}$. Joining these profiles, we represent the casting $\mathcal{C}$ as a height matrix $\mathcal{H}$

$$\mathcal{H} = \begin{pmatrix} h_{1,1} & h_{1,2} & ... & h_{1,m-1} & h_{1,m} \\ h_{2,1} & h_{2,2} & ... & h_{2,m-1} & h_{2,m} \\ & & & & \\ & & ... & & \\ h_{\ell-1,1} & h_{\ell-1,2} & ... & h_{\ell-1,m-1} & h_{\ell-1,m} \\ h_{\ell,1} & h_{\ell,2} & ... & h_{\ell,m-1} & h_{\ell,m} \end{pmatrix} \quad (1)$$

where each $h_{x,y}$ represent the height of the point in the space (x, y). Therefore, the number of profiles of each casting depends on its size.

Once the system has computed the matrix $\mathcal{H}$, we have to remove the possible existing noise, as well as the data unrelated to the casting surface, like the working table. To this end, we establish a height threshold.

We have also taken into consideration other representations of the data using the height matrix:

- **Grey-scale Height Map [9]:** This well-known representation converts each height value to a range between 0 and 255, showing the different scales of grey.
- **Colour Height Map:** This representation is similar to the Grey-scale Height Map but with higher detail, since full colour information is used; specifically, the RGB (i.e., Red, Green and Blue) components. The colour scale we used is Jet Colour Map as defined in Matlab [10].
- **Normal map:** As the former ones, this representation has been generated by means of the height matrix, but shows the direction of the normal vector of the surface for each point in the matrix. Each vector for each point have three components (x, y, z) and we represent them as an image — even it is not a true image — corresponding red component to the x value, green component to the y value and blue component to the z value.

Thenceforth, the system starts with the segmentation process. We define segmentation as the process to select the regions of the surface of the casting that may have defects. To this end, we made two different labelling tasks. The first one classifies each casting as good or defective. This classification is necessary for the construction of the models that are going to be used in the segmentation process. The second one is for evaluating the accuracy of the segmentation and the defect categorisation methods and labels each segment in 'correct', 'inclusion', 'cold lap' and 'misrun'.

The segmentation process is accomplished by the following steps:

1) The system converts to grey-scale the normal map — although it is not an image, we represent it as one, as aforementioned – of the casting and the normal map of the correct models. This step is performed to remove noise of the rugosity of the surface.
2) The Gaussian Blur filter [11] is applied.
3) The systems applies a difference filter between the normal map to examine and each of the correct models.
4) The system applies a intersection filter between the differences computed in the previous step.
5) The system binarizes the results.
6) We apply a algorithm for extracting the areas or segments (potentially faulty), removing the ones which are excessively small.

Once each area is extracted , we computed several features using the different representations. These features can be divided into the following categories:

- **Features of the segmented image:** The segmented image is the result of the segmentation process applied to the normal map.
  - The width, height and perimeter of the area.
  - The euclidean distance of the center of gravity of the area to origin of coordinate axes.
  - The fullness, which is computed as $Area/(Width * Height)$
- **Features of the integral image of segmented binary image:** These features are obtained from the conversion to the integral image of the segmented version of the image. An integral image is defined as the image in which the intensity at a pixel position is equal to the sum of the intensities of all the pixels above and to the left of that position in the original image [12].
  - Mean value of pixels in the integral image.
  - The result of addition of the pixels values in the integral image.
- **Features of the grey scale height map:** They are extracted from the computed segments in the original grey-scale height map.
  - Max, min, mean, median, standard deviaton and entropy of the grey histogram values.
  - Max, min, mean, median, standard deviaton and entropy of the grey histogram values without black (zero value) pixels.
- **Features of the colour height map:** These features are extracted from the computed segments in the original colour height map.
  - Max, min, mean, median, standard deviaton and entropy of the red histogram values.
  - Max, min, mean, median, standard deviaton and entropy of the red histogram values without black (zero value) pixels.
  - Max, min, mean, median, standard deviaton and entropy of the green histogram values.
  - Max, min, mean, median, standard deviaton and entropy of the green histogram values without black (zero value) pixels.
  - Max, min, mean, median, standard deviaton and entropy of the blue histogram values.
  - Max, min, mean, median, standard deviaton and entropy of the blue histogram values without black (zero value) pixels.
- **Features of the normal map:** These features are extracted from the computed segments in the original normal map.
  - Max, min, mean, median, standard deviaton and entropy of the x component histogram values.
  - Max, min, mean, median, standard deviaton and entropy of the x component histogram values without black (zero value) pixels.
  - Max, min, mean, median, standard deviaton and entropy of the y component histogram values.
  - Max, min, mean, median, standard deviaton and entropy of the y component histogram values without

black (zero value) pixels.
- Max, min, mean, median, standard deviaton and entropy of the z component histogram values.
- Max, min, mean, median, standard deviaton and entropy of the z component histogram values without black (zero value) pixels.

Using these features for each segment, we can train machine-learning algorithms in order to categorise them into 4 possible categories: cold lap, inclusion, misrun and correct.

IV. Machine Learning Classifiers

Machine-learning is an active research area within *Artificial Intelligence* (AI) that focuses on the design and development of new algorithms that allow computers to reason and decide based on data [13].

Machine-learning algorithms can commonly be divided into three different types depending on the training data: supervised learning, unsupervised learning and semi-supervised learning. For supervised algorithms, the training dataset must be labelled (e.g., the defect in the casting) [14]. Unsupervised learning algorithms try to determine how data are organised into different groups named clusters. Therefore, data do not need to be labelled [15]. Finally, semi-supervised machine-learning algorithms use a mixture of both labelled and unlabelled data in order to build models, improving the accuracy of solely unsupervised methods [16].

Because castings can be properly labelled, we use supervised machine-learning; however, in the future, we would also like to test unsupervised methods for automatic categorisation of foundry defects.

A. Bayesian Networks

Bayesian Networks [17], which are based on the *Bayes Theorem*, are defined as graphical probabilistic models for multivariate analysis. Specifically, they are directed acyclic graphs that have an associated probability distribution function [18]. Nodes within the directed graph represent problem variables (they can be either a premise or a conclusion) and the edges represent conditional dependencies between such variables. Moreover, the probability function illustrates the strength of these relationships in the graph [18].

The most important capability of Bayesian Networks is their ability to determine the probability that a certain hypothesis is true (e.g., the probability of a casting to have certain defect) given a historical dataset.

B. Decision Trees

Decision Tree classifiers are a type of machine-learning classifiers that are graphically represented as trees. Internal nodes represent conditions regarding the variables of a problem, whereas final nodes or leaves represent the ultimate decision of the algorithm [19].

Different training methods are typically used for learning the graph structure of these models from a labelled dataset. We use *Random Forest*, an ensemble (i.e., combination of weak classifiers) of different randomly-built decision trees [20], and *J48*, the WEKA [21] implementation of the *C4.5* algorithm [22].

C. K-Nearest Neighbour

The *K-Nearest Neighbour* (KNN) [23] classifier is one of the simplest supervised machine learning models. This method classifies an unknown specimen based on the class of the instances closest to it in the training space by measuring the distance between the training instances and the unknown instance.

Even though several methods to choose the class of the unknown sample exist, the most common technique is to simply classify the unknown instance as the most common class amongst the K-nearest neighbours.

D. Support Vector Machines (SVM)

SVM algorithms divide the n-dimensional space representation of the data into two regions using a *hyperplane*. This hyperplane always maximises the *margin* between those two regions or classes. The margin is defined by the farthest distance between the examples of the two classes and computed based on the distance between the closest instances of both classes, which are called *supporting vectors* [24].

Instead of using linear hyperplanes, it is common to use the so-called *kernel functions*. These kernel functions lead to non-linear classification surfaces, such as polynomial, radial or sigmoid surfaces [25].

V. Empirical Validation

In order to evaluate our casting defect detector, we collected a dataset from a foundry, which is specialised in safety and precisions components for the automotive industry (principally, in disk-brake support with a production over 45,000 tons a year). The experiments were focused in three different surface defects:

1) Inclusion.
2) Cold Lap.
3) Misrun.

To construct the dataset, we analysed 645 foundry castings with the segmentation machine-vision system described in Section III in order to retrieve the different segments and their features. In particular, we use 176 correct casting to construct the model and the remainder for testing. By means of this analysis, we construct an dataset of 5785 segments to train machine-learning models and categorise the defects. Besides, we added a fourth category to identify the noise that our machine vision system retrieves called 'Correct', which represent the segments gathered by the segmentation method that are correct even though the method has marked them as potentially faulty. In particular, Table I shows the number of segments in each category.

As it can be noticed, the dataset was not balanced for the fourth existing classes due to scarce data. To address both problems (scarce and unbalanced data) we applied Synthetic Minority Over-sampling TEchnique (SMOTE) [26], which is a combination of over-sampling the less populated classes

TABLE I
NUMBER OF SAMPLES FOR EACH CATEGORY.

Category	*Number of samples*
Inclusion	387
Cold Lap	16
Misrun	52
Correct	5030

and under-sampling the more populated ones. Nevertheless, the over-sampling is performed by creating synthetic minority class examples. In this way, instances were still unique and classes became more balanced.

The acceptance/rejection criterion of the studied models resembles the one applied by the final requirements of the customer. Pieces flawed with defects must be rejected due to the very restrictive quality standards (which is an imposed practice by the automotive industry). To this end, we labelled each possible segment within the castings with its defects.

First of all, we evaluate the coverage of our segmentation method. To this end, we define the metric 'Coverage':

$$Coverage = \frac{S_{s\to s}}{S_{s\to s} + S_{c\to s}} \cdot 100 \qquad (2)$$

where $S_{s\to s}$ is the number of segments retrieved by the segmentation system which are defects and $S_{c\to s}$ are the number of defects that our segmentation method does not gather.

Next, we evaluate the precision of the machine-learning method to categorise the segments. To this extent, by means of the dataset, we conducted the following methodology to evaluate the proposed method:

- **Cross validation:** This method is generally applied in machine-learning evaluation [27]. In our experiments, we performed a K-fold cross validation with $k = 10$. In this way, our dataset is 10 times split into 10 different sets of learning (90 % of the total dataset) and testing (10 % of the total data).
- **SMOTE:** For each training dataset in each fold, we built a dataset that contains the result of applying SMOTE to the original training dataset in order to balance the not balanced classes.
- **Learning the model:** For each fold, we accomplished the learning step of each algorithm using different parameters or learning algorithms depending on the specific model. In particular, we used the following models:
  - *Bayesian networks (BN):* With regards to Bayesian networks, we utilize different structural learning algorithms: K2 [28] and Tree Augmented Naïve (TAN) [29]. Moreover, we also performed experiments with a Naïve Bayes Classifier [27].
  - *Support Vector Machines (SVM):* We performed experiments with a polynomial kernel [25], a normalized polynomial Kernel [30], a Pearson VII function-based universal kernel [31] and a radial basis function (RBF) based kernel [32].
  - *K-nearest neighbour (KNN):* We performed experiments with $k = 1$, $k = 2$, $k = 3$, $k = 4$, and $k = 5$.
  - *Decision Trees (DT):* We performed experiments with J48(the *Weka* [21] implementation of the *C4.5* algorithm [22]) and Random Forest [20], an ensemble of randomly constructed decision trees. In particular, we tested random forest with a variable number of random trees N, $N = 10$, $N = 25$, $N = 50$, $N = 75$, and $N = 100$.

  To test the approach, we evaluated the percent of correctly classified instances and the area under the ROC curve, which establishes the relation between false negatives and false positives [33].

Regarding the coverage results, our segmentation system is able to detect a 59.22% of the faulty castings. More accurately, the defects that the system classifies incorrectly are little inclusions and inclusion in the border of the castings. This coverage value is rather low and, therefore, the segmentation method should be improved to make it higher.

TABLE II
RESULTS OF THE CATEGORISATION IN TERMS OF ACCURACY AND AUC. THE BEST RESULTS WERE OBTAINED BY THE RANDOM FOREST TRAINED WITH MORE THAN 50 TREES.

Model	**Accuracy (%)**	**AUC**
Bayes K2	93.53	0.82
Bayes TAN	92.47	0.82
Naïve Bayes	73.84	0.91
SVM: Polynomial Kernel	89.26	0.92
SVM: Normalised Polynomial Kernel	89.86	0.91
SVM: Pearson VII Kernel	95.95	0.90
SVM: Radial Basis Function Kernel	80.76	0.89
KNN K = 1	95.24	0.86
KNN K = 2	95.77	0.91
KNN K = 3	95.28	0.93
KNN K = 4	95.37	0.93
KNN K = 5	94.89	0.94
J48	93.16	0.83
Random Forest N = 10	95.73	0.95
Random Forest N = 25	95.92	0.96
Random Forest N = 50	95.95	0.97
Random Forest N = 75	96.04	0.97
Random Forest N = 100	96.12	0.97

If we focus in the precision of the categorisation of the segments, Table II shows the results of the categorisation phase. In particular, the best results were obtained by the Random Forest trained with more than 50 trees with an accuracy of more than 95% and an AUC of 0.97. SVM trained with a Radial Basis Function kernel was not a good classifier, implying that a radial division of the space is not as feasible as others, because the rest of the SVMs behaved with accuracies higher 95% in the case of Pearson VII and near 90% in the case of the both of the polynomial kernels. Surprisingly, the lazy classifier KNN achieved very high results, ranging from 95.77% to 94.89% of accuracy and from 0.86 to 0.94 or AUC. J48 an average classifier that achieved a poor AUC: 0.83.

Although the results of the categorisation of the surface defect were good enough to implant this system in a real foundry, the coverage of the segmentation method renders the whole system incomplete. The coverage should be improved in order to make this system deployable.

VI. Conclusions

Foundry is an ancient magic-surrounded activity that has evolved to become one of the key pieces of the whole society as we know it. Because foundry supplies key pieces to other important and critical sectors like aeronautic or automotive industries where the tiniest defect may become fatal.

In this paper, we proposed a new system based on machine vision and machine learning in order to detect and categorise defects in the surface of iron castings. This approach starts by retrieving images from the tested castings. Then, the segmentation method identifies all the possible defects within the castings. Finally, machine-learning models are used to classify the possible defect into inclusion, cold lap, misrun or correct. We evaluated our approach in terms of coverage of the proposed segmentation method and precision of the categorisation of the regions. The experimental results showed that, albeit our precision in categorisation is very high, the coverage of the system should be improved.

Future work is oriented in 2 main ways. First, we are going to develop new segmentation methods in order to enhance our coverage results. Second, we will use different features to improve the categorisation process.

VII. Acknowledgements

This work has been partially supported by the Ministry of Science and Innovation of Spain under the program 'Proyectos Singulares y Estratégicos del Programa Nacional de Cooperación Público-Privada 2009', project 'POLIFEMO: POlifemo, Lowering Imperfections in Foundry Enterprise via a Machine-vision Oriented model' and by the Diputación Foral de Bizkaia under the program 'Plan integral de innovación empresarial. Area 5. Tecnología', project '7/12/TK/2009/0005' named 'Desarrollo de la visión artificial orientada a las variables del sistema de gestión integral del proceso de transformación metálica' .

References

[1] A. Mital, M. Govindaraju, and B. Subramani, "A comparison between manual and hybrid methods in parts inspection," *Integrated Manufacturing Systems*, vol. 9, no. 6, pp. 344–349, 1998.

[2] P. Kopardekar, A. Mital, and S. Anand, "Manual, hybrid and automated inspection literature and current research," *Integrated Manufacturing Systems*, vol. 4, no. 1, pp. 18–29, 1993.

[3] O. Silvén, M. Niskanen, and H. Kauppinen, "Wood inspectionwith nonsupervisedclustering," *Machine Vision and Applications*, vol. 13, no. 5, pp. 275–285, 2003.

[4] V. Murino, M. Bicego, and I. Rossi, "Statistical classification of raw textile defects," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 4. IEEE, 2004, pp. 311–314.

[5] F. Pernkopf, "Detection of surface defects on raw steel blocks using bayesian network classifiers," *Pattern Analysis & Applications*, vol. 7, no. 3, pp. 333–342, 2004.

[6] Y. Frayman, H. Zheng, and S. Nahavandi, "Machine vision system for automatic inspection of surface defects in aluminum die casting," *Journal of advanced computational intelligence*, vol. 10, no. 3, pp. 281–286, 2011.

[7] S. Kalpakjian and S. Schmid, *Manufacturing engineering and technology*, 2005.

[8] F. Pernkopf and P. O'Leary, "Image acquisition techniques for automatic visual inspection of metallic surfaces," *NDT & E International*, vol. 36, no. 8, pp. 609–617, 2003.

[9] D. vom Stein, "Automatic visual 3-d inspection of castings," *Foundry Trade Journal*, vol. 180, no. 3641, pp. 24–27, 2007.

[10] R. Gonzalez, R. Woods, and S. Eddins, *Digital image processing using MATLAB*. Pearson Education India, 2004.

[11] R. Gonzalez and R. Woods, "Digital image processing. 1992," *Reading, Mass.: Addison-Wesley*, vol. 16, no. 716, p. 8.

[12] P. Viola and M. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[13] C. Bishop, *Pattern recognition and machine learning*. Springer New York., 2006.

[14] S. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," in *Proceeding of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, 2007, pp. 3–24.

[15] S. Kotsiantis and P. Pintelas, "Recent advances in clustering: A brief survey," *WSEAS Transactions on Information Science and Applications*, vol. 1, no. 1, pp. 73–81, 2004.

[16] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised learning*. MIT Press, 2006.

[17] J. Pearl, "Reverend bayes on inference engines: a distributed hierarchical approach," in *Proceedings of the National Conference on Artificial Intelligence*, 1982, pp. 133–136.

[18] E. Castillo, J. M. Gutiérrez, and A. S. Hadi, *Expert Systems and Probabilistic Network Models*, erste ed., New York, NY, USA, 1996.

[19] J. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[20] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[21] S. Garner, "Weka: The Waikato environment for knowledge analysis," in *Proceedings of the 1995 New Zealand Computer Science Research Students Conference*, 1995, pp. 57–64.

[22] J. Quinlan, *C4. 5 programs for machine learning*. Morgan Kaufmann Publishers, 1993.

[23] E. Fix and J. L. Hodges, "Discriminatory analysis: Nonparametric discrimination: Small sample performance," *Technical Report Project 21-49-004, Report Number 11*, 1952.

[24] V. Vapnik, *The nature of statistical learning theory*. Springer, 2000.

[25] S. Amari and S. Wu, "Improving support vector machine classifiers by modifying kernel functions," *Neural Networks*, vol. 12, no. 6, pp. 783–789, 1999.

[26] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 3, pp. 321–357, 2002.

[27] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[28] G. F. Cooper and E. Herskovits, "A bayesian method for constructing bayesian belief networks from databases," in *Proceedings of the 1991 conference on Uncertainty in artificial intelligence*, 1991.

[29] D. Geiger, M. Goldszmidt, G. Provan, P. Langley, and P. Smyth, "Bayesian network classifiers," in *Machine Learning*, 1997, pp. 131–163.

[30] S. Maji, A. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *Proc. CVPR*, vol. 1, no. 2, 2008, p. 4.

[31] B. Üstün, W. Melssen, and L. Buydens, "Visualisation and interpretation of support vector regression models," *Analytica chimica acta*, vol. 595, no. 1-2, pp. 299–309, 2007.

[32] B. Cho, H. Yu, J. Lee, Y. Chee, I. Kim, and S. Kim, "Nonlinear support vector machine visualization for risk factor analysis using nomograms and localized radial basis function kernels," *IEEE Transactions on Information Technology in Biomedicine*, vol. 12, no. 2, p. 247, 2008.

[33] Y. Singh, A. Kaur, and R. Malhotra, "Comparative analysis of regression and machine learning methods for predicting fault proneness models," *International Journal of Computer Applications in Technology*, vol. 35, no. 2, pp. 183–193, 2009.