Corrosion Detection for Automated Visual Inspection

Francisco Bonnin-Pascual, Alberto Ortiz

1. Introduction

Vessels constitute one of the most cost effective forms of transporting bulk goods around the world. However, despite the efforts on reducing maritime accidents, they still occur and, from time to time, have catastrophic consequences both in personal, environmental and financial terms. Structural failure is the major cause of ships wreckages and, as such, Classification Societies impose extensive inspection schemes for assessing the structural integrity of vessels.

The external and internal parts of the hull can be affected by different kinds of defects typical of steel surfaces and structures, being corrosion of paramount importance. Nowadays, to detect these defects, visual hull inspections are carried out at a great cost [1]: the vessel has to be emptied and situated in a dockyard, where typically temporary staging, lifts, and/or movable platforms need to be installed to allow the workers for close-up inspection (and repair if needed) of all the different metallic surfaces and structures. Taking into account the huge dimensions of some vessels, this process can mean the visual assessment of more than 600,000 m^2 of steel. Besides, the surveys are on many occasions performed in hazardous environments for which the access is usually difficult, while the operational conditions turn out to be sometimes extreme for human operation. For large tonnage vessels, such as Ultra Large Crude Carriers (ULCC), total expenses can be as high as one million euros.

Corrosion is a clear indicator of the state of the hull metallic structures, and, thus, it is of great interest for the surveyor [2]. Different kinds of corrosion may arise: *general corrosion*, that appears as non-protective friable rust which can occur uniformly on uncoated surfaces; *pitting*, a localized process that is normally initiated due to local breakdown of coating and that derives, through corrosive attack, in deep and relatively small diameter pits that can in turn lead to hull penetration in isolated random places; *grooving*, again a localized process, but this time characterized by linear shaped corrosion which occurs at structural intersections where water collects and flows; and *weld metal corrosion*, which affects the weld deposits, mostly due to galvanic action with the base metal, and are likelier in manual welds than in machine welds.

The goal of the EU-funded FP7 MINOAS project [3] is to reengineer the inspection process through the incorporation of robotic technologies. Some of these robots are equipped with cameras which can provide images of the different areas of the vessel hull to be inspected [1]. This chapter revises related work in visual general defect detection, including corrosion, and then describes two pattern recognition approaches developed within the context of the MINOAS project to detect corrosion from digital images.

2. Related Work

Talking about automated visual defect detection, the scientific literature contains an important number of proposals. These can be classified depending on the kind of surface in which they are looking for defects: some approaches face the defect detection on particular objects or surfaces (e.g. LCD displays [4], printed circuit boards [5], copper strips surfaces [6], ceramic tiles [7], etc) while other methods are intended for the detection of defects in generic surfaces (e.g. [8–11]).

A second classification can be established depending on the kind of defect that they try to detect. In this regard, many approaches intended for the inspection of generic surfaces look for general and unspecific defects, although there is an important amount of contributions that are dedicated to a specific type of defect. This is the case of [12], [13] and [14], which are part of a large collection of contributions to automated visual crack detection.

Regarding corrosion detection from images, just a few works can be found in the literature (see e.g. [15–18]). This reduced amount of contributions indicates that there is still much work to do. By way of example, the following sections introduce two novel algorithms for corrosion detection and assess their performance against a varied set of test images.

3. Weak-classifier Colour-based Corrosion Detector (WCCD)

3.1. Description of the algorithm

WCCD is a supervised classifier which has been built around a cascade scheme, although its two stages can be considered as weak classifiers. The idea is to chain different fast classifiers with poor performance in order to obtain a global classifier attaining a much better global performance. To this end, each weak classifier takes profit from different features of the items to classify, reducing the number of false positive detections at each stage. For a good global performance, the classifiers must present a false negative percentage close to zero.

The first stage of the cascade is based on the premise that a corroded area presents a rough texture. Roughness is then related to the energy of the symmetric *gray-level co-occurrence-matrix* (GLCM), calculated for downsampled intensity values between 0 and 31, for a given direction α and distance d [19]. The energy is obtained by means of Equation 1:

$$E = \sum_{i=0}^{31} \sum_{j=0}^{31} p(i,j)^2 \,, \tag{1}$$

where $p(i,j)$ is the probability of the occurrence of gray levels i and j at distance d and orientations α or $\alpha + \pi$. Patches with an energy lower than a given threshold τ_E, i.e. exhibit a rough texture, are finally candidates to be more deeply inspected.

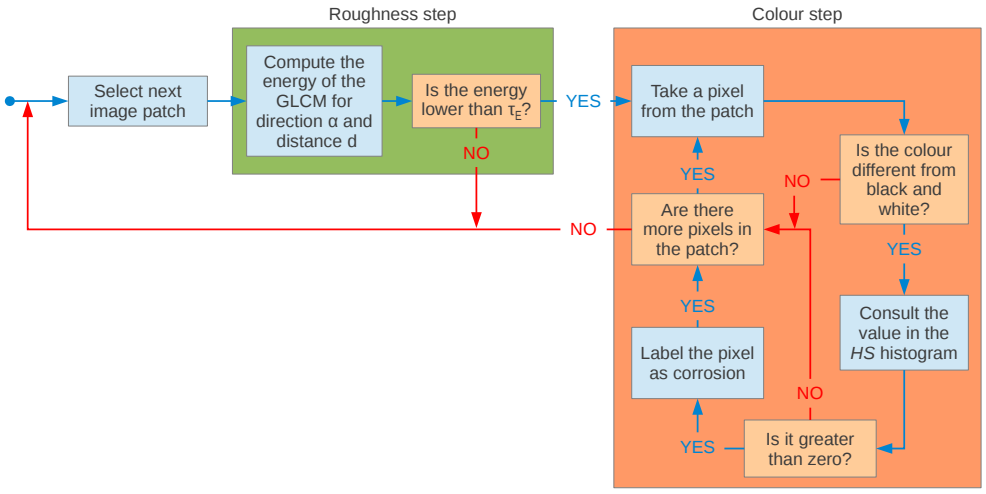

Figure 1. WCCD algorithm flowchart

The second stage filters the pixels of the patches that have passed the roughness stage. This stage makes use of the colour information that can be observed from corroded areas. More precisely, the classifier works over the Hue-Saturation-Value (HSV) space after the realization that HSV-values that can be observed in corroded areas are confined in a bounded subspace of the HS plane. Although the V component has been observed neither significant nor necessary to describe the color of corrosion, it is used to prevent the well-known instabilities in the computation of hue and saturation when color is close to white or black. In that case, the pixel is classified as non-corroded.

A training step is performed prior to the application of this second stage of the corrosion classifier. In this case, training consists of building a bi-dimensional histogram of HS values for image pixels known to be affected by corrosion in the training image set. The resulting histogram is subsequently filtered by zeroing entries whose value is below 10% the highest peak.

The classifier proceeds as follows for every 3-tuple (h, s, v):

(1) pixels close to black, $v < mV$, or white, $v > MV \wedge s < mS$, are labeled as non-corroded, and

(2) for the remaining pixels, the HS histogram is consulted and the pixel is labelled as corroded if $HS(h, s) > 0$,

for given thresholds mV, MV and mS.

Notice that the stages of the cascade cannot be reversed since they do not work with the same kind of entities: while the second stage works at the pixel level, the first stage operates over 15×15 -pixel image patches since it depends on texture, which necessarily involves a pixel neighborhood. Figure 1 shows the flow diagram of WCCD.

3.2. Performance of WCCD

The performance of WCCD depends on the performance of its different stages. To configure the parameters of the roughness stage, several experiments have been performed considering different values for d and α when calculating the GLCM and, consequently, its energy level. No significant differences have been observed among the output values, and so the parameter values are set to $d = 5$ (pixels) and $\alpha = 0$ (horizontal direction). As for the energy threshold τ_E, its value determines the algorithm performance in terms of computation time and number of false positives, since all patches with a high energy level are discarded and only those with a low value become input of the colour checking step.

The parameters of the colour-based step, mV, MV and mS, are set to prevent the instabilities of h and s values from affecting the pixel classification. Using 8-bit HSV values, the *minimum value* mV is set to 50, as well as the *minimum saturation* mS. The *maximum value* MV is set to 200.

Figure 2 provides classification outputs for the same input image using different energy thresholds τ_E. This parameter can be tuned to decrease false positives and just allow the detection of the most significant corroded areas. In the images, pixels labelled as corroded are colour-coded to indicate the probability of successful classification. To be more precise, the colour depends on the height of the corresponding histogram bin in the following way:

- red if $HS(h,s) \in [0.75\mathcal{HS}, 1.00\mathcal{HS}]$,
- orange if $HS(h,s) \in [0.50\mathcal{HS}, 0.75\mathcal{HS}]$,
- green if $HS(h,s) \in [0.25\mathcal{HS}, 0.50\mathcal{HS}]$ and
- blue if $HS(h,s) \in [0.10\mathcal{HS}, 0.25\mathcal{HS}]$,

where $\mathcal{HS} = \max\{HS(\cdot,\cdot)\}$.

The performance of the detector has been quantified by means of a varied set of test images and manually generated ground truth data (after the proper configuration of the different parameters, as explained above). False positive and false negative percentages, respectively *FP/no. pixels* and *FN/no. pixels*, have been used as the figures of merit. The process carried out to perform this assessment has entailed the implementation of different techniques to filter the HS histogram and a posterior comparison among the alternatives with regard to the generalization capability of the resulting classifier.

Downsampling the histogram to 32 levels for hue and saturation has been the first filter considered, which merely groups bins with similar hue-saturation values. As can be seen in the first and second rows of Table 1, this filter has resulted in a considerable improvement in comparison with the original 256×256 HS histogram, thus it has been considered as the reference for comparing with the other filtering strategies.

More specifically, two more attempts have been performed in order to reduce the false negative percentage while preserving the false positive percentage. On the one hand, the Parzen windows method [19] has been applied to the original 256×256 histogram using the two-dimensional Gaussian kernel shown in Equation 2:

$$G(x,y,\sigma) = \frac{1}{2\pi\,\sigma^2}\; e^{-\frac{1}{2}\left(\frac{(x-\mu_x)^2}{\sigma^2} + \frac{(y-\mu_y)^2}{\sigma^2}\right)}, \tag{2}$$

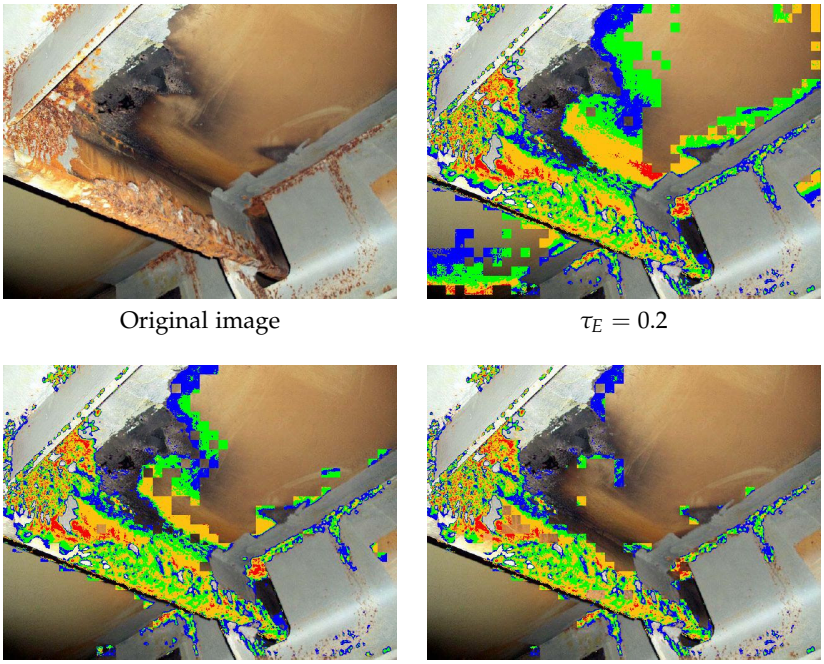

Figure 2. Corroded areas detected by WCCD for different energy threshold values

where μ_x and μ_y are the hue and saturation values for the neighbourhood center, x and y are the values for a nearby sample, and σ is the standard deviation. The algorithm performance has been assessed filtering the histogram using different values for σ. Best misclassification rates and percentages are shown in the third row of Table 1, which correspond to $\sigma = 12$.

On the other hand, a Bilateral filter [20] has been applied to the original 256×256 histogram, considering the bins height as the intensity values of an image. This approach filters the histogram using a kernel consisting of two Gaussians, one for the spatial domain and another for the range domain. After the different experiments carried out, the best performance has been obtained for $\sigma_{spatial} = 15$, $\sigma_{range} = 1$ and a kernel size of 30 pixels. The fourth row of Table 1 provides the resulting performance values. The resulting histograms for the different filters can be found in Figure 3. By way of conclusion, it seems the approach based on the bilateral filter is the one providing best results, although it is true the different strategies lead to a final similar performance. The bilateral filter has thus been selected for being part of WCCD.

Examples of final classification outputs for WCCD are provided in Figure 4.

Regarding the execution times, WCCD provides corrosion-labelled images in 7-25 ms. These execution times correspond to images ranging from 120.000 to 172.000 pixels, and for a runtime environment comprising a laptop fitted with an Intel Core2 Duo processor (@2.20GHz, 4GB RAM).

Table 1. Misclassification measures for different *HS* histograms (WCCD)

	FP percentage	FN percentage
Original (256 bins)	0.80	13.56
Downsampled to 32 bins	9.78	6.10
Using Parzen windows	9.23	5.99
Using a Bilateral filter	9.80	5.86

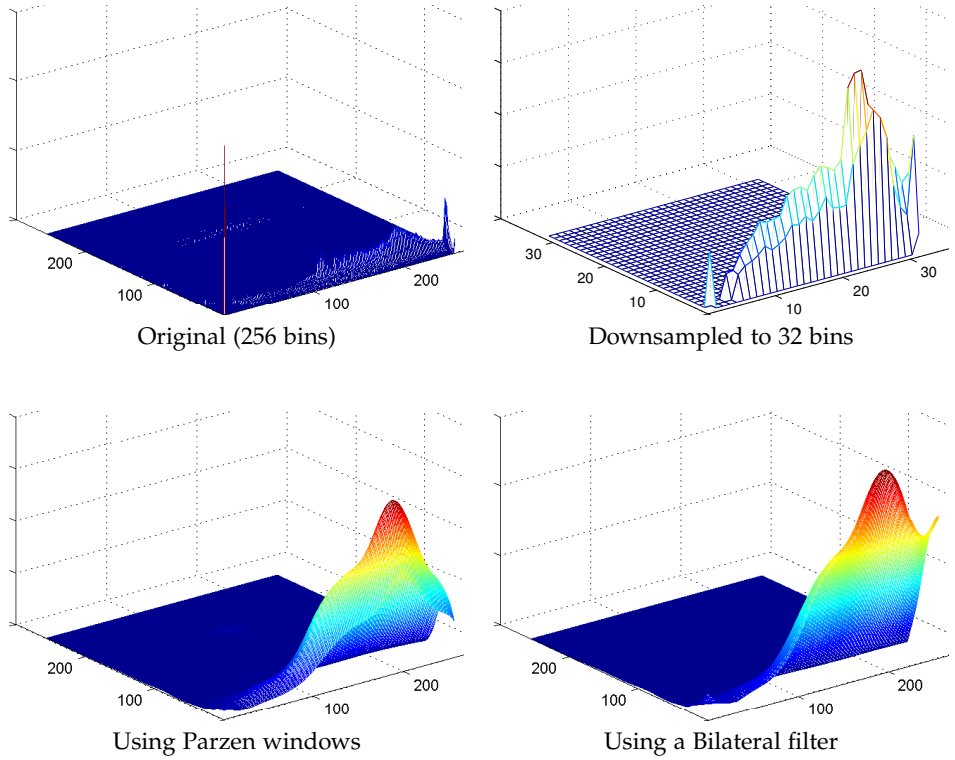

Figure 3. HS histograms resulting from the different filtering strategies (WCCD)

4. AdaBoost based Corrosion Detector (ABCD)

4.1. Description of the algorithm

ABCD makes use of the *Adaptive Boosting* paradigm (AdaBoost) for both learning and classifying corroded areas. Decision trees, as produced by the *Classification and Regression Trees* (CART) learning technique [21, 22], are used as weak classifiers.

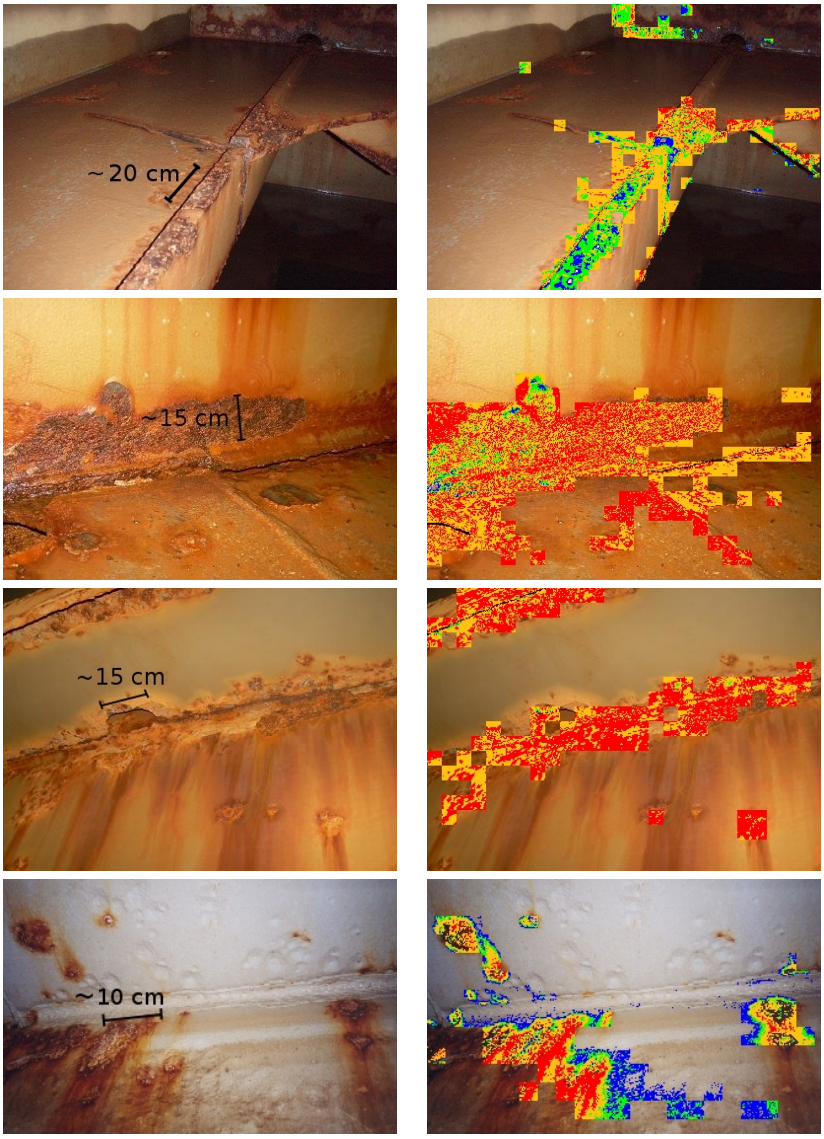

Figure 4. (1st column) Test images and (2nd column) corroded areas detected by WCCD

Laws' texture energy filter responses are used to feed the decision trees. This is so because these filters are able to enhance different features of every material texture. We use a filter bank with 48 different filters that are obtained after combining the following five 1D five-component basic filters:

level L5 = [1 4 6 4 1] edge E5 = [-1 -2 0 2 1]
spot S5 = [-1 0 2 0 -1] wave W5 = [-1 2 0 -2 1]
ripple R5 = [1 -4 6 -4 1]

To describe a texture, the corresponding gray-level patch is convolved with the set of energy filters ($T \otimes filter \rightarrow c$) and different statistical measures are taken over a 15×15 neighbourhood of the filter response, which finally constitute the texture descriptor:

$$\mu = \frac{\sum_{u=0,v=0}^{N,M} \mid c(u,v) \mid}{NM} \tag{3}$$

$$\sigma = \sqrt{\sum_{u=0,v=0}^{N,M} \frac{(c(u,v)-\mu)^2}{NM}} \tag{4}$$

$$\mu^{+} = \frac{\sum_{u,v \mid c(u,v)>0}^{N,M} c(u,v)}{NM} \tag{5}$$

$$\mu^{-} = \frac{\sum_{u,v \mid c(u,v)<0}^{N,M} c(u,v)}{NM} \tag{6}$$

During the learning process, AdaBoost is fed with 192 statistical measures per image patch (4 statistical measures $\times$ 48 energy filters), together with the roughness of the patch (computed as in the first stage of the WCCD detector) and its class label. The output is a set of weak classifiers together with their weights, what allows a correct discrimination of those patches which belong to defective areas from those which does not.

In addition, the same technique used in the second stage of the WCCD algorithm, based on a Hue-Saturation histogram, is also used in ABCD to filter the pixels from the patches that are classified as corroded by AdaBoost. The flow diagram for ABCD is shown in Figure 5.

4.2. Performance of ABCD

We have considered three versions of AdaBoost: *Real*, *Gentle* and *Modest*. *Real AdaBoost* is the generalization of a basic AdaBoost algorithm. *Gentle AdaBoost* [23] is a more robust and stable version of Real AdaBoost, used, for example, by the Viola-Jones object detector. Finally, *Modest AdaBoost* [24] is a version mostly aimed for better resistance to overfitting.

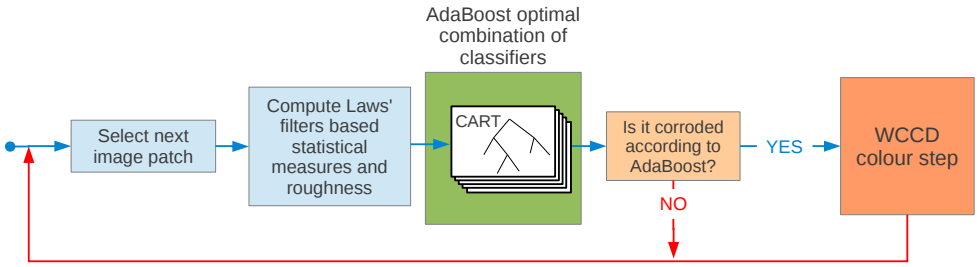

Figure 5. ABCD algorithm flowchart

Table 2. Error percentage obtained for the different AdaBoost versions

Real	Gentle	Modest
6.05	5.99	9.52

For training the algorithm, a total number of 39746 patches have been gathered from 25 different images. These patches have been labelled as *defective* (12952 patches) or *non-defective* (26794) by means of visual inspection. One half of the total amount of patches has been used to train the different versions of AdaBoost, while the other half has been used as control samples to assess the performance of the resulting classifiers.

The AdaBoost parameters have been configured as follows:

1. the maximum number of boosting iterations, i.e. the number of weak classifiers that make up the final classifier, has been set to 100.
2. the tree depth, that is, the depth of the CART, which determines how good the weak classifiers are, has been set to three levels.

Both parameters have been configured to improve the detection performance without prolonging the learning time unnecessarily.

After the execution of the three versions of AdaBoost, their performances have been assessed. Table 2 shows the error percentages obtained for the three versions.

As can be seen, best results are obtained for the Gentle version of AdaBoost. Some results obtained for this version, after adding the colour-based filter, are shown in Figure 6. Among the different images shown, special mention is done for the image in the first row, where corrosion in form of *pitting*, affecting a very reduced fraction of the image, is successfully detected.

The final performance of the complete algorithm has been analysed following the same procedure used for assessing the performance of WCCD. The false positive and false negative percentages have been again used as the figures of merit. The values obtained are shown in Table 3.

Regarding the execution times, ABCD provides corrosion-labelled images in 300-512 ms. These execution times correspond to the same images and runtime environment used for assessing WCCD.

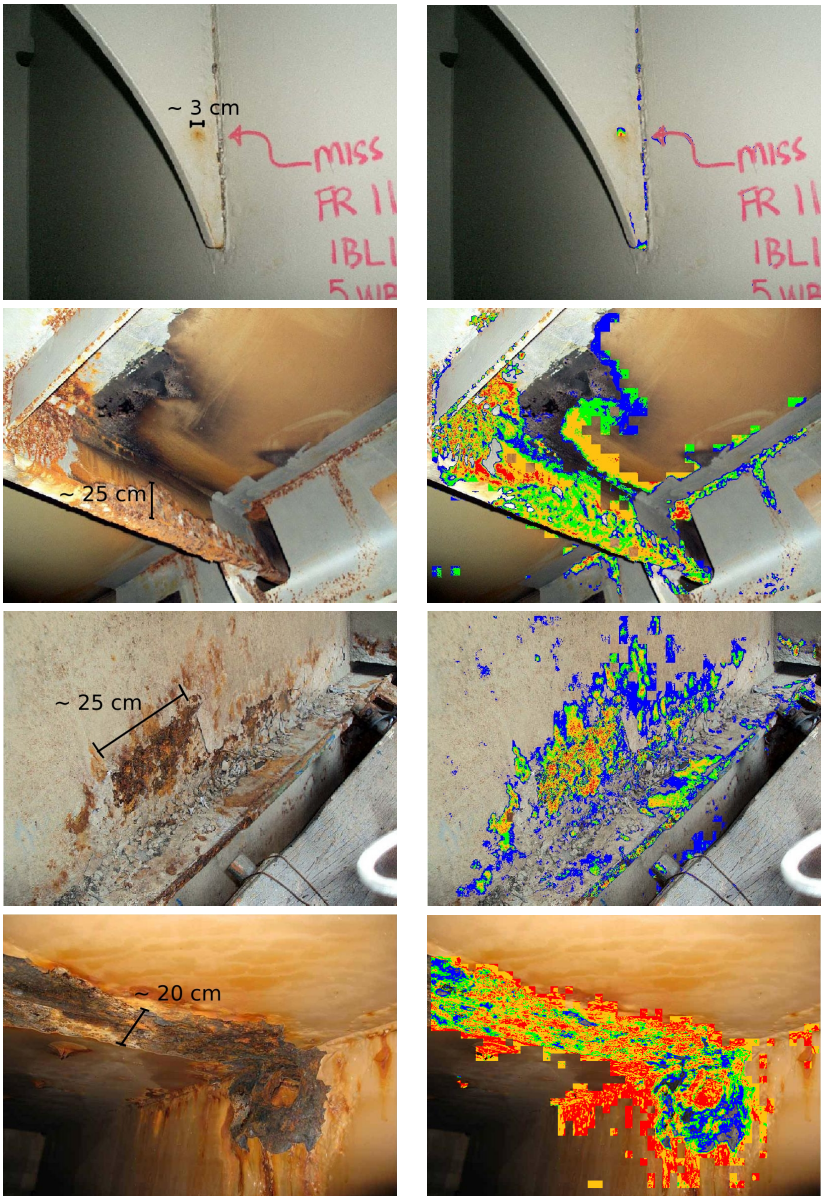

Figure 6. (1st column) Test images and (2nd column) corroded areas detected by ABCD

Table 3. Misclassification measures for ABCD

FP %	FN %
17.16	3.39

5. Conclusions

In this chapter we have introduced two vision-based corrosion detection algorithms that have been developed within the context of the European project MINOAS. Both algorithms are based on the idea of combining weak classifiers for obtaining a good global performance.

After assessing their performance, the misclassification percentages obtained for both algorithms result to be not null. These results can be explained by analysing the kind of misclassifications and the areas where they appear. On the one hand, the FN percentages are not zero because the detectors tend to label as corrosion the center of the corroded area, while the borders are usually not totally labelled. On the other hand, the FP percentages are neither null due to the presence of different structures in the image that are misclassified as defects.

Nevertheless, if corroded areas are considered as entities and it is assumed that the labelling of a single pixel within a defective area is useful, then the ratio between the number of undetected defective areas and the true number of defective areas turns out to be zero for both algorithms, since all them are always detected. In this regard, it is important to remember that the detectors are intended to be used to facilitate visual inspections, and, thus, reporting about the existence of corroded areas in images is considered worth enough even if the areas are not completely labelled.

Taking into account the misclassification ratios, it is not clear whether WCCD performs better or worse than ABCD. However, based on the shorter execution times and the qualitative evaluation of the results, it seems that WCCD outperforms ABCD.

Acknowledgements

This work has been partially supported by FP7 projects MINOAS (GA 233715) and INCASS (GA 605200) and by the European Social Found through grant FPI10-43175042V (Conselleria d'Educacio, Cultura i Universitats, Govern de les Illes Balears).

Author details

Francisco Bonnin-Pascual[1],*, Alberto Ortiz[1]

* Address all correspondence to: xisco.bonnin@uib.es

[1] Department of Mathematics and Computer Science, University of Balearic Islands, 07122 Palma de Mallorca, Spain

6. References

[1] A. Ortiz, F. Bonnin-Pascual, A. Gibbins, P. Apostolopoulou, W. Bateman, M. Eich, F. Spadoni, M. Caccia, and L. Drikos. First steps towards a roboticized visual inspection system for vessels. In *Proc. of the IEEE Emerging Technologies and Factory Automation conference*, 2010.

[2] MINOAS. D1 - definition of the inspection plan / definition of acceptance criteria. In *MINOAS Public Deliverable*, 2011. http://minoasproject.eu/excibition/Publications /PDF/D1.pdf.

[3] M. Caccia, R. Robino, W. Bateman, M. Eich, A. Ortiz, L. Drikos, A. Todorova, I. Gaviotis, F. Spadoni, and V. Apostolopoulou. Minoas a marine inspection robotic assistant: system requirements and design. In *Proceedings of the IFAC Symposium on Intelligent Autonomous Vehicles*, 9 2010.

[4] C.-P. Hsu C.-L. Chang, H.-H. Chang. An intelligent defect inspection technique for color filter. In *Proceedings of the IEEE International Conference on Mechatronics*, pages 933–936, 2005.

[5] B. C. Jiang, C. C. Wang, and P. L. Chen. Logistic regression tree applied to classify pcb golden finger defects. *The International Journal of Advanced Manufacturing Technology*, 24(7-8):496–502, 2004.

[6] X. Zhang, R. Liang, Y. Ding, J. Chen, D. Duan, and G. Zong. The system of copper strips surface defects inspection based on intelligent fusion. In *Proceedings of the IEEE International Conference on Automation and Logistics*, pages 476–480, 2008.

[7] C. Boukouvalas, J. Kittler, R. Marik, M. Mirmehdi, and M. Petrou. Ceramic tile inspection for colour and structural defects. *Technical Report CS-EXT-1995-052*, 1995. Also: Proceedings of AMPT95, pp. 390-399, 1995.

[8] T. Amano. Correlation based image defect detection. In *Proceedings of the IAPR International Conference on Pattern Recognition*, pages 163–166, 2006.

[9] H. Castilho, J. Pinto, and A. Limas. An automated defect detection based on optimized thresholding. In *Proceedings of the International Conference on Image Analysis and Recognition*, volume 4142 of *Lecture Notes in Computer Science*, pages II:790–801, 2006.

[10] J. Hongbin, Y. Murphey, S. Jinajun, and C. Tzyy-Shuh. An intelligent real-time vision system for surface defect detection. In *Proceedings of the IAPR International Conference on Pattern Recognition*, pages III: 239–242, 2004.

[11] A. Kumar and H. Shen. Texture inspection for defects using neural networks and support vector machines. In *Proceedings of the IEEE International Conference on Image Processing*, pages III:353–356, 2002.

[12] Y. Fujita, Y. Mitani, and Y. Hamamoto. A method for crack detection on a concrete structure. In *Proceedings of the IAPR International Conference on Pattern Recognition*, pages III: 901–904, 2006.

[13] R. Oullette, M. Browne, and K. Hirasawa. Genetic algorithm optimization of a convolutional neural network for autonomous crack detection. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages I:516 – 521, 2004.

[14] T. Yamaguchi and S. Hashimoto. Fast crack detection method for large-size concrete surface images using percolation-based image processing. *Machine Vision and Applications*, 21(5):797–809, 2010.

[15] M.R. Jahanshahi and S.F. Masri. Effect of color space, color channels, and sub-image block size on the performance of wavelet-based texture analysis algorithms: An application to corrosion detection on steel structures. In *ASCE International Workshop on Computing in Civil Engineering*, 2013.

[16] G. Ji, Y. Zhu, and Y. Zhang. The corroded defect rating system of coating material based on computer vision. In *Transactions on Edutainment VIII*, volume 7220 of *Lecture Notes in Computer Science*, pages 210–220. Springer Berlin Heidelberg, 2012.

[17] M. Siegel and P. Gunatilake. Robotic enhanced visual inspection of aircraft skin, 1999.

[18] B. B. Zaidan, A. A. Zaidan, H. O. Alanazi, and R. Alnaqeib. Towards corrosion detection system. *International Journal of Computer Science Issues*, 7(1):33–35, 2010.

[19] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, 3rd Edition*. Academic Press, 2006.

[20] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 839 – 846, 1998.

[21] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth International, 1984.

[22] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, 2001.

[23] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 38(2):337 – 374, 2000.

[24] A. Vezhnevets and V. Vezhnevets. Modest adaboost-teaching adaboost to generalize better. Technical report, Graphicon, 2005.