Visual Inspection of Vessels by means of a Micro-Aerial Vehicle: an Artificial Neural Network Approach for Corrosion Detection

Alberto Ortiz*, Francisco Bonnin-Pascual,
Emilio Garcia-Fidalgo, and Joan P. Company

Dep. Mathem. and Computer Science, Univ. Balearic Islands (Spain)
{alberto.ortiz}@uib.es

Abstract. Periodic visual inspection of the different surfaces of a vessel hull is typically performed by trained surveyors at great cost, both in time and in economical terms. Assisting them during the inspection process by means of mechanisms capable of automatic or semi-automatic defect detection would certainly decrease the inspection cost. This paper describes a defect detection approach comprising: (1) a Micro-Aerial Vehicle (MAV) which is used to collect images from the surfaces under inspection, particularly focusing on remote areas where the surveyor has no visual access; and (2) a coating breakdown/corrosion detector based on a 3-layer feed-forward artificial neural network. The success of the classification process depends not only on the defect detector but also on a number of assistance functions that are provided by the control architecture of the aerial platform, whose aim is to improve picture quality. Both aspects are described along the different sections of the paper, as well as the classification performance attained.

Keywords: Corrosion Detection, MAV, Artificial Neural Network

1 Introduction

The movement of goods by ships is today one of the most time and cost effective methods of transportation. The safety of these vessels is overseen by Classification Societies, who are continually seeking to improve standards and reduce the risk of maritime accidents. Structural failures are a major cause of such accidents, which can be usually prevented through timely maintenance. As such, vessels undergo annual inspections, with intensive Special and Docking Surveys every five years, what ensures that the hull structure and related piping are all in satisfactory condition and are fit for the intended use over the next five years.

* This work is partially supported by the European Social Fund through grant FPI11-43123621R (Conselleria d'Educacio, Cultura i Universitats, Govern de les Illes Balears) and by the EU FP7 project INCASS (GA 605200). This publication reflects only the authors' views and the European Union is not liable for any use that may be made of the information contained therein. The authors also thank Fabricio Ardizon for his involvement in project INCASS.

An important part of vessels maintenance has to do with the visual inspection of the hull, where the surveyor is expected to be within arm's reach to the structure under inspection. As it is well known, the hull can be affected by different kinds of defective situations typical of steel surfaces, such as coating breakdown, corrosion and, ultimately, cracks. These defects are indicators of the state of the metallic surface and, as such, an early detection prevents the structure from buckling and / or fracturing.

To carry out this task, the vessel has to be emptied and situated in a dockyard where scaffolding and / or cherry-pickers must be used to allow the human inspectors to reach the area under inspection. For some vessels (e.g. Ultra Large Crude Carriers, ULCC), 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 operational conditions turn out to be sometimes extreme for human operation. Moreover, total expenses involved by the infrastructure needed for close-up inspection of the hull can reach up to $1M once you factor in the vessel's preparation, use of yard's facilities, cleaning, ventilation, and provision of access arrangements. In addition, vessel owners experience significant lost opportunity costs while the ship is inoperable. It is therefore clear that any level of automation of the inspection process is to lead to a reduction of the inspection time, a reduction of the financial costs involved and/or an increase in the safety of the operation.

One of the main goals of the already concluded EU FP7 project MINOAS was to develop a fleet of robotic platforms with different locomotion capabilities with the aim of teleporting the human surveyor to the different vessel structures to be inspected. Given the enormity of these structures and the requirement for vertical motion as part of the inspection process, a multirotor platform was selected as one of the members of the fleet due to their small size, agility and fast deployment time. These platforms, also named as Micro-Aerial Vehicles (MAVs) by some authors, have become increasingly popular in recent years. In accordance to some constructive advice from end-users at the end of project MINOAS, a re-design of this platform is currently under development within the EU FP7 follow-up project INCASS.

As an additional contribution in this line, this paper presents a novel solution for detecting coating breakdown / corrosion in images as a support for surveyors during vessel inspection. The solution here described adopts an approach based on a 3-layer feed-forward neural network which detects pixels suspected to correspond to defective areas. To improve the success rate, the aforementioned aerial platform has been fitted with a number of autonomous functionalities for enhanced image capture. This is achieved by means of extensive use of behaviour-based high-level control.

The paper is organized as follows: Section 2 reviews previous work on the subject, Section 3 describes the aerial platform that is used for image collection, Section 4 describes the corrosion detection algorithm, Section 5 reports on the result of a number of experiments using vessel surface images, and, finally, Section 6 concludes the paper.

2 Previous Work

Referring to automated visual defect detection, the scientific literature contains an important number of proposals. Among other possibilities, these can be roughly classified in two categories, depending on whether they look for defects specific for particular objects or surfaces —e.g. LCD displays [7], printed circuit boards [17], copper strips [27], ceramic tiles [6], etc.— or, to the contrary, they aim at detecting general and unspecific defects –e.g. [1, 4, 8, 14, 18].

Within the first category, one can find a large collection of contributions for automatic vision-based crack detection, mainly for concrete surfaces —e.g. [11, 20, 25]. However, regarding corrosion, apart from some contributions of part of the authors of this paper [3, 5], to the best of our knowledge, the number of works which can be found is rather reduced [15, 16, 22, 24, 26]. First of all, [15] makes use of color wavelet-based texture analysis algorithms for detecting corrosion, while [16] utilizes the watershed transform applied over the gradient of gray-level images, [22] uses wavelets for characterizing and detect corrosion texture in airplanes, [24] adopts an approach based on the fractal properties of corroded surfaces and [26] also focuses on corrosion texture using the standard deviation and the entropy as the discriminating features. In our previous works, we have used texture and colour descriptors through a cascade of weak classifiers [3] and Law's energy filters and Adaboost [5], among others.

3 The Aerial Platform

3.1 General overview

In line with the robotic platform developed for the MINOAS project, the new micro-aerial vehicle is based on a multi-rotor configuration. The control software has been configured to be hosted by any of the research platforms developed by Ascending Technologies (the quadcopters Hummingbird and Pelican, and the hexacopter Firefly), although it could be adapted to other systems. The AscTec vehicles are equipped with an inertial measuring unit (IMU) that comprises a 3-axis gyroscope, a 3-axis accelerometer and a 3-axis magnetometer. Attitude stabilization control loops linked to the onboard IMU and thrust control run over the main ARM7 microcontroller as part of the platform firmware. The manufacturer leaves almost free an additional secondary ARM7 microcontroller which can execute onboard higher-level control loops.

Figure 1 shows a Hummingbird platform configured for the visual inspection application, whose sensor suite comprises:

– Two optical flow sensors, one looking to the ground and the other pointing forward, to estimate the vehicle speed with regard to, respectively, the ground and the front wall (i.e. the surface to be inspected). To this end, we make use of the PX4Flow sensor developed within the PX4 Autopilot project [13]. This sensor comprises a CMOS high-sensitivity imaging sensor, an ARM Cortex M4 microcontroller to compute the optical flow at 250 Hz, a

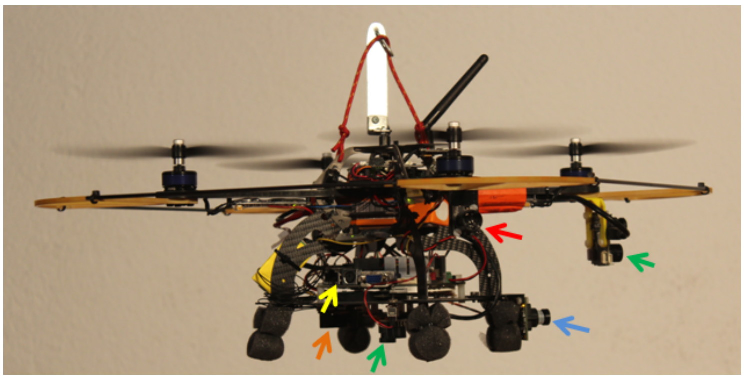

Fig. 1. A Hummingbird platform featuring the visual inspection sensor suite: (green) front-looking and bottom-looking optical flow sensors, (red) ultrasound sensor, (orange) height sensor, (yellow) embedded PC, (blue) camera.

3-axis gyroscope for angular rate compensation, and a MaxBotix ultrasonic (US) sensor HRLV-EZ4, with 1 mm resolution, used to scale the optical flow to metric velocity values. The distance measured by the US sensor is an additional output supplied by this device.

– Range sensors pointing in different directions for obstacle detection and collision prevention. Currently, we use two Maxbotix HRLV-EZ4 sensors oriented to the left and to the right. This kind of sensor provides information at 10 Hz and its detection range is up to 5 m.
– An additional range sensor which provides height estimation when flying above the 5 m covered by the PX4Flow oriented downwards. To this end, we use the Teraranger One [21], which consists in an infrared time-of-flight measurement sensor that provides an extended range of 14 m.
– A set of cameras that collect the requested images from the vessel structures under inspection. The specific configuration depends on the inspection to perform and the payload capacity of the platform. For instance, the Hummingbird shown in Fig. 1 features a minimalistic configuration comprising a single forward-looking uEye UI-1221LE camera.

The vehicle carries an additional processing board which avoids sending sensor data to a base station, but process them onboard and, thus, prevent communications latency inside critical control loops. This processor will be referred to as the high-level processor from now on. The configuration shown in Fig. 1 comprises a Commell LP-172 Pico-ITX board featuring an Intel Atom 1.86 GHz processor and 4 GB RAM.

3.2 Control Software

The aerial platform implements a control architecture that follows the *supervised autonomy* (SA) paradigm [9]. This is a human-robot framework where the robot implements a number of autonomous functions, including self-preservation and other safety-related issues, which make simpler the intended operations for the user, so that he/she, which is allowed to be within the general platform control loop, can focus in accomplishing the task at hand. Within this framework, the communication between the robot and the user is performed via qualitative instructions and explanations: the user prescribes high-level instructions to the platform while this provides instructive feedback. In our case, we use a joystick to introduce the qualitative commands and a *graphical user interface* (GUI) to receive the robot feedback. Joystick commands and the GUI are handled at a *base station* (BS) linked with the MAV via a WiFi connection.

In more detail, the control software has been organized around a layered structure distributed among the available computational resources. On the one hand, as said above, the *low-level control* layer implementing attitude stabilization and direct motor control executes over the main microcontroller as the platform firmware provided by the manufacturer [12]. On the other hand, *mid-level control*, running over the secondary microcontroller, comprises height and velocity controllers which map input speed commands into roll, pitch, yaw and thrust orders. Lastly, the *high-level control* layer implements a reactive control strategy coded as a series of ROS nodes running over Linux Ubuntu, which combine the user desired speed command with the available sensor data —x, y, z velocities, height z and distances to the closest obstacles—, to obtain a final and safe speed set-point that is sent to the speed controllers.

Speed commands are generated through a set of robot behaviors organized in a hybrid competitive-cooperative framework [2]. That is to say, on the one hand, higher priority behaviors can overwrite the output of lower priority behaviors by means of a suppression mechanism taken from the *subsumption* architectural model. On the other hand, the cooperation between behaviors with the same priority level is performed through a *motor schema*, where all the involved behaviors supply each a motion vector and the final output is their weighted summation. An additional flow control mechanism selects, according to a specific input, between the output provided by two or more behaviours.

Figure 2 details the behavior-based architecture, grouping the different behaviors depending on its purpose. A total of four general categories have been identified for the particular case of visual inspection: (a) *behaviors to accomplish the user intention*, which propagate the user desired speed command, attenuating it towards zero in the presence of close obstacles, or keeps hovering until the WiFi link is restored after an interruption; (b) *behaviors to ensure the platform safety within the environment*, which prevent the robot from colliding or getting off the safe area of operation, i.e. flying too high or too far from the reference surface that is involved in optical flow measurements; (c) *behaviors to increase the autonomy level*, which provide higher levels of autonomy to both simplify the vehicle operation and to introduce further assistance during inspections; and (d) *behaviors to check flight viability*, which checks whether the flight can start or progress at a certain moment in time. Some of the behaviors in groups (a) and (c) can operate in the so-called *inspection mode.* While in this mode, the vehicle

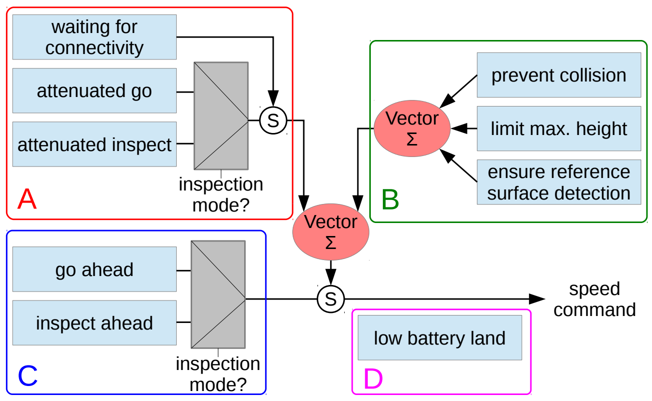

Fig. 2. MAV behaviors: A – behaviors to accomplish the user intention, B – behaviors to ensure the platform safety within the environment, C – behaviors to increase the autonomy level, and D – behaviors to check flight viability.

moves at a constant and reduced speed (if it is not hovering) and user commands for longitudinal displacements or turning around the vertical axis are ignored. In this way, during an inspection, the platform keeps at a constant distance and orientation with regard to the front wall, for improved image capture.

4 Artificial Neural Network for Corrosion Detection

This section describes a coating breakdown/corrosion (CBC) detector based on a *multi-layer perceptron* configured as a *feed-forward neural network* (FFNN), which discriminates between the CBC and the NC (non-corrosion) classes.

4.1 Background

An *artificial neural network* (ANN) is a computational paradigm that consists of a number of units (neurons) which are connected by weighted links (see Fig. 3). This kind of computational structure learns from experience (rather than being explicitly programmed) and is inspired from the structure of biological neural networks and their way of encoding and solving problems. An FFNN is a class of ANN which organizes neurons in several layers, namely one input layer, one or more hidden layers, and one output layer, in such a way that connections exist from one layer to the next, never backwards [23], i.e. recurrent connections between neurons are not allowed. Arbitrary input vectors propagate forward through the network, finally causing an activation vector in the output layer. The entire network function, which maps input vectors onto output vectors, is determined by the connection weights of the net w_{ij}.

Every neuron k in the network is a simple processing unit that computes its activation output o_k with respect to its incoming excitation $x = \{x_i \,|\, i =$

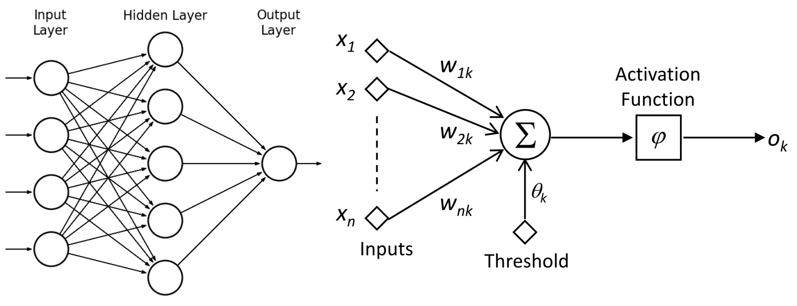

Fig. 3. (left) Topology of a simple FFNN comprising one single hidden layer. (right) Structure of an artificial neuron.

$1, \ldots, n\}$, in accordance to $o_k = \varphi\left(\sum_{i=1}^{n} w_{ik} x_i + \theta_k\right)$, where φ is the so-called activation function, which, among others, can take the form of e.g. the hyperbolic tangent $\varphi(z) = 2/(1 + e^{-az}) - 1$. Training consists in tuning weights w_{ik} by optimizing the summed square error function $E = 0.5 \sum_{p=1}^{N} \sum_{j=1}^{r} (o_j^p - t_j^p)^2$, where N is the number of training input patterns, r is the number of neurons at the output layer and (o_j^p, t_j^p) are the current and expected outputs of the j-th output neuron for the p-th training pattern. Taking as a basis the *back-propagation algorithm*, a number of alternative training approaches have been proposed through the years, such as the *delta-bar-delta rule*, *QuickpPop*, *Rprop*, etc [10].

4.2 Network Configuration

In order to determine an optimal setup for the classifier, we have considered a number of plausible combinations and performed tests accordingly in order to adopt the best configuration. First of all, we have defined both color and texture descriptors to characterize the neighbourhood of each pixel. Next, we have also considered different structures for the NN varying the number of hidden neurons. In detail:

– As for color, we have checked the *hue* and *saturation* channels of the HSV color space, as well as the r and g channels of normalized RGB. Furthermore, two kinds of descriptors have been considered for both cases: (1) the average value of each channel within a neighbourhood, and (2) stacked 8-bin histograms for downsampled intensity values for each channel in the same neighbourhood. In the former case, the descriptor comprises two components, while the latter configuration results in a total of $8 + 8 = 16$ components.
– Regarding texture, center-surround differences have been considered in the form of: (1) signed *surrounding differences* (SD) between a central pixel and its neighbourhood, and (2) 10-bin histograms of *uniform local binary patterns* (LBP) [19]. See Fig. 4 for an illustration of both descriptors for a distance of one pixel between the neighbourhood $\{n_i\}$ and the central pixel p. During testing, the 2-pixel distance case has also been taken into account.

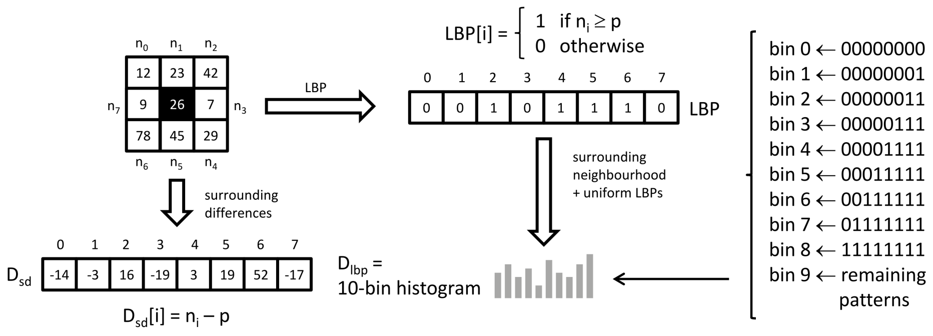

Fig. 4. Illustration of the *surrounding differences* (D_{sd}) and *uniform LBP* (D_{lbp}) texture descriptors (in the latter case, to achieve rotational invariance, every bin accounts not only for the indicated pattern but also for the corresponding rotations).

– Finally, the number of hidden neurons have been varied from 0.5 to 2.5 times the number of components of the input pattern (in 0.1 increments).

All in all, a total of 2 (texture descriptors) $\times$ 2 (1-pixel & 2-pixel distances) $\times$ 2 (color descriptors) $\times$ 2 (average/histogram) $\times$ 21 (hidden neuron quantities) = 336 combinations have been considered. Since every configuration was trained with 5 different initializations (to avoid dependence in this regard), the total number of tests finally performed was 1680. All layers make use of the hyperbolic tangent as activation function.

5 Experimental Results

Figure 5 represents, in TPR-FPR space, the performance of the full set of 336 configurations, where TPR is the *true positive rate* [TPR = TP/(TP+FN)] and FPR is the *false positive rate* [FPR = FP/(TN+FP)], where a *positive* represents membership to the CBC class. In the TPR-FPR space, the perfect classifier lies at the (0,1) point.

Among all classifiers, those whose performance lie closer to the (0,1) point are clearly preferrable to those ones that are farther. Table 1 shows, in this regard, the average and standard deviations of those distances for every combination of descriptors, measured over a total of $21 \times 5 = 105$ trainings, varying the number of hidden neurons as specified above. As can be observed, the best combination results to be LBP histograms at 1-pixel distance for texture and 8-bin histograms of (h,s) channels for color (case T2/R1/C1/H8), which exhibits the shortest average distance to (0,1) as well as the smallest standard deviation, what indicates little variation in global performance among the different trainings and amounts of hidden neurons (from $0.5 \times (10+8+8) = 13$ to $2.5 \times 26 = 65$ neurons). During this selection, all neighbourhoods were set to 15×15 pixels for all descriptors.

Table 2(left) provides the details for the previous combination of descriptors and the best result achieved, which corresponds to the case of 37 hidden

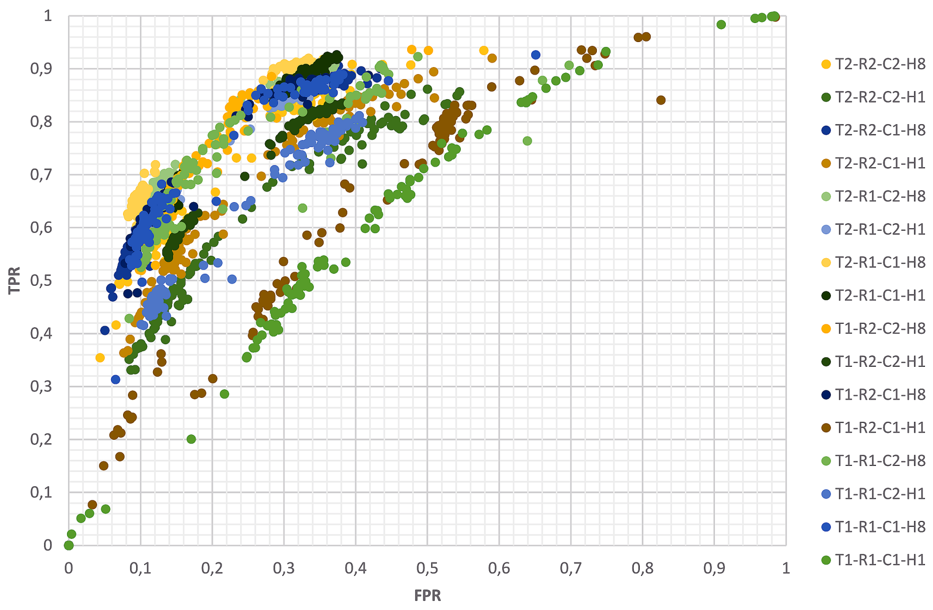

Fig. 5. FPR *versus* TPR for all combinations of descriptors.

neurons, while the results contained in Table 2(middle) are for the same configuration but after tuning the descriptors with additional training rounds. Improved performance was attained after combining LBP and SD descriptors, and after varying the neighbourhood sizes. The final configuration has resulted to be: 10-bin histograms of uniform LBPs at 1-pixel distance measured over 17×17-pixel windows, 8 surrounding differences at 1-pixel distance, 8-bin histograms of hue values and 8-bin histograms of saturation values, both measured over 11 × 11-pixel values, and a 3-layer FFNN, with 10+8+8+8=34-component input vectors and 37 neurons in the hidden layer.

The activation function employed –i.e. the hyperbolic tangent– makes the output of the neural network to be always a value between -1 and 1, with -1 corresponding to the NC class and 1 corresponding to the CBC class. Typically, the current pattern should be classified as CBC if its output value is closer to 1 than to -1. However, output values around 0 are likely not to lead to correct classifications. After some final tests, patterns for which the output values lie within the [-0.6,+0.6] interval were decided to be left unclassified. Furthermore, in a further post-processing stage, the classification output was median-filtered (1 × 15-pixel window), in order to avoid misclassification results due to image noise. These final enhancements have led to a 90.71% success rate, for a total of 13.71% unclassified pixels. See Table 2(right) for the details.

To finish, Figure 6 shows classification results for some of the images of the dataset employed for the performance assessment.

Table 1. Average and standard deviation of distances to point (0,1) for all combinations of descriptors. (T1 = SD, T2 = LBP, Ri = i-pixel distance, C1 = (h,s) color channels, C2 = (r,g) color channels, H1 = average, H8 = 8-bin histogram.)

Combination	C1 / H1	C1 / H8	C2 / H1	C2 / H8
T1 / R1	0.73 (0.16)	0.40 (0.09)	0.47 (0.06)	0.39 (0.07)
T1 / R2	0.64 (0.12)	0.36 (0.05)	0.41 (0.03)	0.35 (0.06)
T2 / R1	0.35 (0.02)	**0.33 (0.02)**	0.38 (0.04)	0.34 (0.02)
T2 / R2	0.47 (0.07)	0.43 (0.07)	0.51 (0.07)	0.42 (0.08)

Table 2. Performance details for the best combination of descriptors.

T2/R1/C1/H8		truth					
		CBC	NC	CBC	NC	CBC	NC
classification output	CBC	656,680	3,661,374	644,853	1,769,771	543,199	1,087,733
	NC	91,092	9,841,962	110,165	11,726,319	53,768	10,611,366
success rate		0.74		0.87		0.91	
TPR		0.88		0.85		0.91	
FPR		0.27		0.13		0.09	
distance to (0,1)		0.30		0.20		0.13	
		before tuning		*after tuning*		*no doubtful pixels*	

6 Conclusions and Future Work

A Micro-Aerial Vehicle to be used for vessel visual inspection has been presented, together with a corrosion detection approach based on an artificial neural network. The MAV control approach is based on the SA paradigm, and hence the user is introduced in the platform control loop. For the specific problem of visual inspection, we have proposed and fully described a behaviour-based control architecture which include among its functionalities the enhancement of image capture. Regarding the corrosion detection approach, the classifier building process has been described and successful detection results have been reported, with a success rate of around 91% among the different performance metrics. Next steps for improving corrosion detection performance include enhancing the navigation capabilities of the MAV (and hence get higher quality images), and reduce false positives by means of a general defects pre-detection stage, so that the ANN corrosion detector is mostly fed with defective area images.

References

1. Amano, T.: Correlation based image defect detection. In: Proc. ICPR. pp. 163–166 (2006)

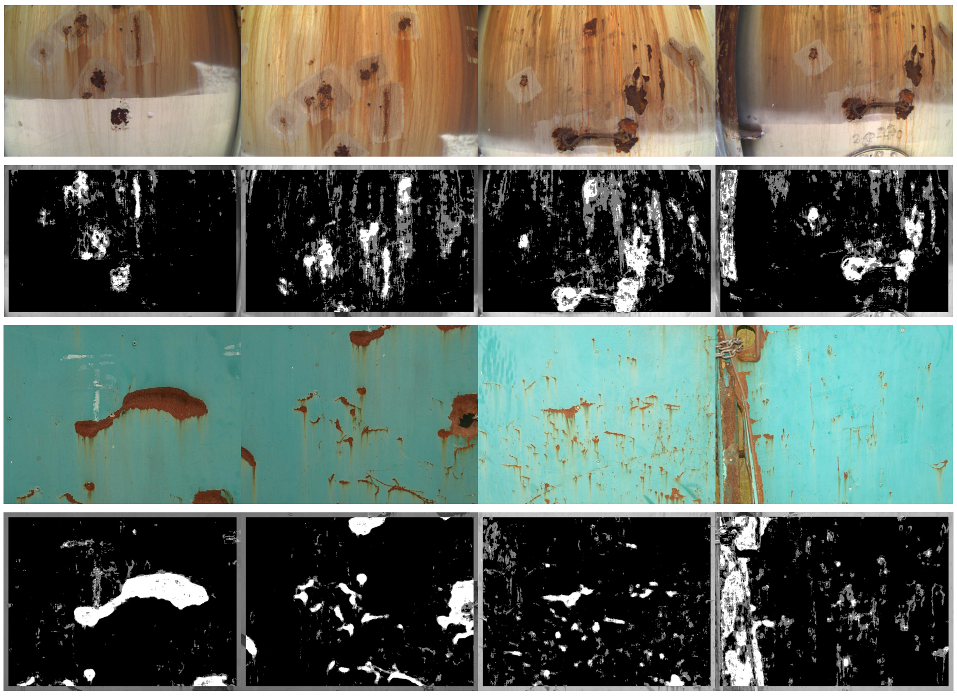
Fig. 6. Classification results for some of the images of the dataset: white - CBC class, black - NC class, gray - unclassified.

2. Arkin, R.C.: Behavior-based Robotics. MIT press (1998)
3. Bonnin-Pascual, F., Ortiz, A.: Combination of Weak Classifiers for Metallic Corrosion Detection and Guided Crack Location. In: IEEE Intl. Conf. Emerging Tech. & Factory Automat. (2010)
4. Bonnin-pascual, F., Ortiz, A.: A Probabilistic Approach for Defect Detection Based on Saliency Mechanisms. In: IEEE Intl. Conf. Emerging Tech. & Factory Automat. (2014)
5. Bonnin-Pascual, F., Ortiz, A.: Corrosion Detection for Automated Visual Inspection. In: Aliofkhazraei, D.M. (ed.) Developments in Corrosion Protection, chap. 25, pp. 619–632. InTech (2014)
6. Boukouvalas, C., Kittler, J., Marik, R., Mirmehdi, M., Petrou, M.: Ceramic tile inspection for colour and structural defects. Technical Report CS-EXT-1995-052 (1995), also: Proceedings of AMPT95, pp. 390-399, 1995
7. C.-L. Chang, H.-H. Chang, C.P.H.: An intelligent defect inspection technique for color filter. In: Pro. IEEE Intl. Conf. Mechatronics. pp. 933–936 (2005)
8. Castilho, H., Pinto, J., Limas, A.: An automated defect detection based on optimized thresholding. In: Proc. Intl. Conf. Image Analysis & Recognition. pp. 790–801 (2006)
9. Cheng, G., Zelinsky, A.: Supervised Autonomy: A Framework for Human-Robot Systems Development. Auton. Robot. 10, 251–266 (2001)
10. Duda, R., Hart, P., Stork, D.: Pattern Classification. Wiley (2001)

11. Fujita, Y., Mitani, Y., Hamamoto, Y.: A method for crack detection on a concrete structure. In: Proc. ICPR. pp. III: 901–904 (2006)
12. Gurdan, D., Stumpf, J., Achtelik, M., Doth, K.M., Hirzinger, G., Rus, D.: Energy-efficient Autonomous Four-rotor Flying Robot Controlled at 1 kHz. In: Proc. IEEE ICRA. pp. 361–366 (2007)
13. Honegger, D., Meier, L., Tanskanen, P., Pollefeys, M.: An Open Source and Open Hardware Embedded Metric Optical Flow CMOS Camera for Indoor and Outdoor Applications. In: Proc. IEEE ICRA. pp. 1736–1741 (2013)
14. Hongbin, J., Murphey, Y., Jinajun, S., Tzyy-Shuh, C.: An intelligent real-time vision system for surface defect detection. In: Proc. ICPR. vol. III, pp. 239–242 (2004)
15. Jahanshahi, M., Masri, S.: 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 Intl. Workshop on Computing in Civil Eng. (2013)
16. Ji, G., Zhu, Y., Zhang, Y.: The corroded defect rating system of coating material based on computer vision. In: Trans. Edutainment VIII, LNCS, vol. 7220, pp. 210–220. Springer (2012)
17. Jiang, B.C., Wang, C.C., Chen, P.L.: Logistic regression tree applied to classify PCB golden finger defects. Intl. Jour. Advanced Manufacturing Technology 24(7-8), 496–502 (2004)
18. Kumar, A., Shen, H.: Texture inspection for defects using neural networks and support vector machines. In: Proc. IEEE ICIP. vol. III, pp. 353–356 (2002)
19. Ojala, T., Pietikäinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. Pattern Recognit. 29(1), 51–59 (1996)
20. Oullette, R., Browne, M., Hirasawa, K.: Genetic algorithm optimization of a convolutional neural network for autonomous crack detection. In: Proc. IEEE CEC. pp. I:516 – 521 (2004)
21. Ruffo, M., Castro, M.D., Molinari, L., Losito, R., Masi, A., Kovermann, J., Rodrigues, L.: New Infrared Time-of-flight Measurement Sensor for Robotic Platforms. In: IMEKO TC4 Int. Symposium and Int. Workshop on ADC Modelling and Testing. pp. 13–18 (2014)
22. Siegel, M., Gunatilake, P., Podnar, G.: Robotic assistants for aircraft inspectors. IEEE Instrumentation & Measurement Magazine 1(1), 16–30 (1998)
23. Theodoridis, S., Koutroumbas, K.: Pattern Recognition. Academic Press (2006)
24. Xu, S., Weng, Y.: A new approach to estimate fractal dimensions of corrosion images. Pattern Recogn. Lett. 27(16), 1942–1947 (2006)
25. Yamaguchi, T., Hashimoto, S.: Fast Crack Detection Method for Large-size Concrete Surface Images Using Percolation-based Image Processing. Mach. Vision Appl. 21(5), 797–809 (2010)
26. Zaidan, B.B., Zaidan, A.A., Alanazi, H.O., Alnaqeib, R.: Towards corrosion detection system. Intl. Jour. Computer Science Issues 7(1), 33–35 (2010)
27. Zhang, X., Liang, R., Ding, Y., Chen, J., Duan, D., Zong, G.: The system of copper strips surface defects inspection based on intelligent fusion. In: Proc. IEEE Intl. Conf. Automation and Logistics. pp. 476–480 (2008)