
Automated Defect Detection for Fluorescent Penetrant Inspection using Random Forest

N.J. Shipway^{a,*}, T.J. Barden^b, P. Huthwaite^a, M.J.S. Lowe^a

^a*Department of Mechanical Engineering, Imperial College London, Exhibition Road, London, SW7 2AZ, UK*

^b*NDE Laboratory, Rolls-Royce plc., Bristol BS34 7QE, UK*

Abstract

Fluorescent Penetrant Inspection (FPI) is the most widely used NDT method in the aerospace industry. Inspection of FPI is currently done visually and difficulties arise distinguishing between penetrant associated with defects and that due to insufficient wash-off or geometrical indications. This, in addition to the nature of the inspection process, means inspection is largely influenced by human factors. The ability to perform automated inspection would provide increased consistency, reliability and productivity.

The Random Forest algorithm was used to detect defects in a number of flat titanium plates which had been processed with FPI and photographed to produce digital images. This method has demonstrated the ability to correctly distinguish between defects and other non-relevant indications with accuracy comparable to a human inspector with a very small number of training examples. These results show the potential for the Random Forest algorithm to be used to detect defects in aerospace components, allowing the entire FPI line to become autonomous.

Keywords: Random Forest, Machine Learning, Dye Penetrant, Fluorescent Penetrant Inspection, Automation

1. Introduction

Fluorescent penetrant inspection (FPI) is an established non-destructive testing (NDT) method used extensively in the aerospace industry to detect surface cracks or defects. Whilst automated processing of parts has been common in industry for over 20 years, inspection is still performed manually. The nature of FPI in new manufacture means inspectors can be working lengthy shifts

*Corresponding author at. Department of Mechanical Engineering, Imperial College London, Exhibition Road, London SW7 2AZ, UK. Tel: +44 2075977227

Email address: n.shipway15@imperial.ac.uk (N.J. Shipway)

under low light conditions inspecting identical components with very few defects occurring; typically around 1 in 1000 [32, 31]. These conditions can lead to operator fatigue which in the worst case, could lead to defective parts passing an inspection process undetected. A number of engine failures have occurred as a result of defects being missed during FPI inspection [25, 26, 27], and this has led to the motivation for automated FPI analysis. The introduction of automated processing, as well as increasing consistency and reliability, increased productivity by over 100% compared to a manual system [6]. It is hoped the introduction of automated inspection would further this.

There has been a limited amount of research into the topic of automated defect detection for FPI. The most common approach has been to perform a brightness threshold on the photographic images of the penetrant-treated surfaces, followed by feature extraction which is then used to identify defects. The first attempt at this was work by Tang in 1995 [35]. A threshold method based on boundary shapes was used, followed by pattern classification methods. Artificially induced cracks which were superimposed onto images of engine run components were detected with a correct call rate of 97.7% [35]. Abend [5] then published work on the Opto-tec system which uses a windowing technique to measure and calculate flaw geometries which were then compared to a reference standard. Any deviation from this reference standard would cause the system to stop [5].

Another threshold and feature extraction method was presented by Alba [7]. The features were used to correctly identify defects in the range of 8 - 20 mm as either round or linear. Zheng presented a solution which uses adaptive thresholding and then proceeds to extract features such as centroid, length, width, area etc [36]. From these features, the indication can be classified.

Whilst previous work demonstrates advances towards the goal of automated defect detection for FPI, there are still three challenges which need to be overcome.

The first is the need for a solution which is capable of detecting small defects (approximately 0.75mm). Under optimum conditions, FPI is capable of detecting defects as small as 0.75mm without the use of magnification. In order to produce a system capable of replicating the performance of a human operator, an automated inspection system should be able to identify defects as small, if not smaller than the operator can, else the system will be redundant as all parts will need to be inspected visually.

The second challenge is the ability to provide a system which can identify defects despite variability in size and shape. When defects occur in aerospace components they vary in shape and size. A report by the US Federal Aviation Administration outlines the forty-plus factors relating to FPI which can

affect the results of an inspection [21]. With so many factors affecting results, it is essential the system is robust to variation in indication shape, size, brightness, texture, etc.

The third is the need for the ability to distinguish between real defects and false indications. False indications can arise for a number of reasons. These include insufficient wash-off, geometrical indications, scratches or other roughness on the surface and dust or other foreign objects which can cause fluorescence on the surface. It is common for false indications to appear on the surface of components and it is therefore required that the system should correctly identify defects from false indications. It is for this reason simpler defect detection systems, such as those which use e.g. thresholding, shape/length information, or brightness information alone are unsuccessful. Being able to distinguish between penetrant associated with defects from that associated with other false indications will required a more sophisticated approach.

This work uses the Random Forest technique, proposed by Breiman [12], to identify penetrant associated with small defects from that caused by false indications. It has successfully detected small defects with indications varying in shape and size.

Section 2 will outline underlying theory behind Random Forest and Fluorescent Penetrant Inspection. The methodology is presented in Section 3. Section 4 presents the results, before discussing conclusions and future work in Section 5.

2. Theory

2.1. Fluorescent Penetrant Inspection

FPI is one of the most widely used NDT methods in the aerospace industry, with over 90% of metallic components receiving at least one FPI during production [11, 23]. Its prominence is largely due to the ability to easily and cheaply inspect large complex geometries. It can also be used on a wide variety of materials provided the surface is not too rough or porous [17].

The method of performing FPI involves first cleaning the surface of the component. A fluorescent dye is then applied and after an allocated dwell time, the dye will have seeped into any surface breaking defects via capillary action. After the dwell time has elapsed, the dye is washed off either using water or an emulsifier depending on the penetrant sensitivity and the component is then dried. After the component has dried a developer is applied to the surface, which encourages dye contained within defects to bleed out creating an indication which is much larger and easier to detect than the flaw itself.

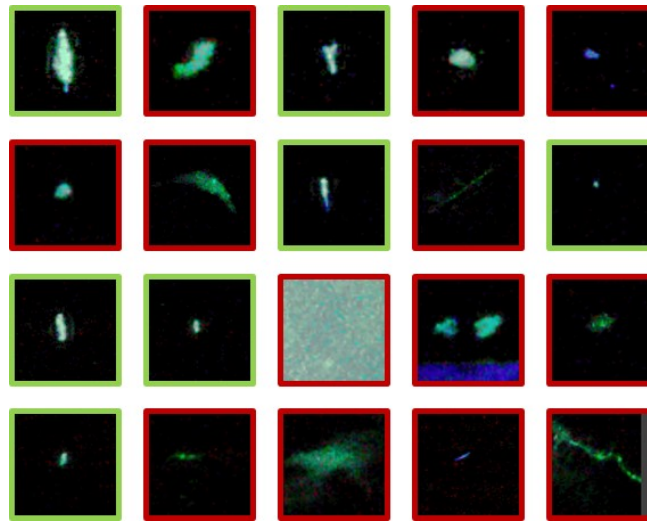


Figure 1: Examples of defects (green border) and false indications (red border)

To inspect the part, the component is placed under UV light and after an eye adaptation time has elapsed, the inspector can proceed to inspect. Inspectors are trained to systematically inspect the component, considering each potential indication, such as those shown in Figure 1. The wipe off technique is a method used by inspectors to assist in distinguishing defects. Acetone is wiped over the indication using a cotton bud or similar. Should the indication reappear, the inspector can be confident a defect is present as the reappearance of the indication signals penetrant is contained within a defect, however should the indication disappear, it is probably not associated with a defect.

The high value and criticality of aerospace components means NDT is of utmost importance to aerospace companies. There is a growing drive towards using NDT as a process control rather than the traditional testing of finished products. Components are inspected multiple times throughout manufacture. This shift has led to demand for more automation to make it easier to analyse NDT data quickly so it can be fed back to the manufacturing process to rectify problems and reduce waste [19].

FPI is also used for other types of component inspection such as repair and overhaul and in-situ. In the case of repair and overhaul, the chance of defects occurring is much higher as the components have been in service. Engine run components have also been found to produce significantly more background penetrant making inspection more complex due to the increased number and variation of false indications.

On the other hand, the use of in-situ inspection requires local processing which creates inconsistencies including variation in the background. Moreover, imaging with a borescope leads to differing

image types due to varying subject to camera distance, viewing angle and illumination.

The initial application for this work will therefore be inspections where variability is tightly controlled, for example surface roughness, processing and imaging. This will ensure spurious signals are minimised and it is hoped this will improve performance. This is typically found in new manufacture, where a repeatable environment can be produced. For these reasons, it is unlikely that either repair and overhaul or in-situ inspection will be the first application for an automated inspection system.

2.2. Machine Learning

Machine learning is a rapidly growing field in which systems learn to solve a task by being shown examples together with the desired outputs for each case. The problem considered here is a supervised classification task in which the aim is to identify which class (normal/defect) a particular image, or image patch, belongs. The system is trained by showing it a large number of image patches from normal examples and patches around defects. The system then learns which image features are important and how to combine them to decide the class of a new patch.

There are many established methods for supervised classification learning problems such as Logistic Regression [16], Artificial Neural Networks (ANN) [22] and Support Vector Machines (SVM) [15, 33].

The Random Forest algorithm was first proposed by Breiman in 2001 [12] and is a development on the established decision tree method [28]. The Random Forest builds a number of decision trees which each have an element of randomness introduced. This randomness is traditionally introduced by using a randomly selected subset of the total number of training examples, with replacement, and also by randomly selecting a subset of the total number of available features. Once the Random Forest has been built, a new example is presented. Each tree in the Random Forest then votes for a classification. The classification with the most number of votes is then used as the predicted classification.

Random Forest has been found to be effective as it has accuracy as good, if not better than the Adaboost method [12], a similar decision tree technique without random elements. It is robust to outliers and noise and it is faster than other decision tree methods such as bagging or boosting [12]; the Law of Large Numbers also means over-fitting does not tend to occur for Random Forest [12]. Other advantages of the Random Forest method are that it is easy to follow the logic taken [10], easy to train [14, 10], fast to train [9] and it has been demonstrated to be effective even with a small number of training examples (around 500 training examples were used by Chan et al. to train around 1 - 700 trees with 1 - 10 input features [14]). These attributes mean the method is well suited to the problem of automated defect detection.

Successful implementation of machine learning methods normally arises out of having a large number of examples with which the system is trained [4, 8]. However, this shall be more challenging for automated defect detection of FPI. The current method of visually inspecting components does not obtain images of components. Also, using FPI as a process control means few defective parts occur. The combination of these two factors mean there is not a large data set of images of FPI processed components containing defects which can be used to train the system. A key outcome of this paper will therefore be to assess the performance of the Random Forest technique with a limited training dataset.

3. Experimental Method

In order to provide a suitable dataset from which the system can be trained, and also to provide a geometrically simple shape to trial with, it was decided that an available set of samples that had been made for a Probability Of Detection (POD) trial, here referred to as POD test plates, would be used as the initial application. In this case, the POD test plates are rectangular titanium plates (100 mm×50 mm×4 mm) and each have a number of cracks of known size artificially induced at random locations using thermal fatigue loading. 36 POD plates were used to conduct this work with 124 cracks ranging from 0.1 mm to 3.4 mm long.

The parts were processed using ultra high sensitivity penetrant (1D4 [1]) and photographed using a Canon EOS 6D DSLR camera with an EF 100 mm f/2.8 USM macro lens. This camera was selected as it offers a high dynamic range. High dynamic range is needed as the majority of the subject is under low light; however the regions of interest, the penetrant, are very bright. The EOS 6D also comes with a full-frame CMOS sensor which minimises noise. The EF 100mm f/2.8 allows a large reproduction ratio (the largest you can reproduce a subject on the sensor compared to its real life size) whilst being positioned at a relatively large working distance. This means the camera can be positioned at a sufficient distance so that it does not interfere with the UV light hitting the surface of the component. The camera was mounted on a stand, at a distance of 426 mm from the table (422 mm from the surface of the component). The UV light used was a Spectroline Tritan 365 and this was positioned around 380 mm from the surface of the component. Each time images were taken, a light meter was used to ensure the UV and ambient light levels were within the accepted range as specified by the Rolls-Royce FPI specification, RRP58003 [29]. Other camera settings are shown in Table 1.

A calibration piece, W1351, was used each time FPI was performed, to ensure consistency. W1351

Property	Value
F-stop	f/4
Exposure time	1/160 sec
ISO speed	ISO 100
Exposure bias	0 step
Focal length	100 mm
White balance	5200 K

Table 1: Camera settings for data acquisition

is shown in Figure 2. There are three cracks along the central axis, sizes 0.7 mm, 1.7 mm and 3.0 mm. Rolls-Royce specification for FPI, RRP58000 [30], states that an inspector is expected to detect defects above 0.75 mm for flat machined samples. Using the calibration piece, it is therefore necessary all defects are detected. The exposure time of the camera was determined according to the brightest region on the calibration test piece. The brightest region is normally the penetrant associated with the 3.0 mm crack. Figure 2a shows the image of W1351 taken at the specified exposure time and Figure 2b shows that image after altering the gamma correction. It should be noted Figure 2b has only been included to allow the reader to better see W1351. This image is not used by the defect detection software at all. More examples of processed plates are shown in Figure 3. These have been included to demonstrate the typical level of background penetrant visible.

Each image was annotated manually using a graphics editor to indicate which pixels are associated with defects. This creates a mask image, see Figure 2c, which provides ground truth information to the defect detection software to learn from. The first stage of the defect detection software considers each image and detects pixels above a brightness level of 1000, see Figure 2d. (Each pixel is a 16-bit integer and therefore the maximum brightness level is 65,535.) This focuses the locations considered by the software to regions containing penetrant. Windows are formed around each of these bright locations, producing windows such as those shown in Figure 4. The size of the windows is varied and the location of each is recorded to allow a prediction image to be constructed at the end of the analysis based on whether the software determines that particular window to be defective or not. All other locations which fall below the brightness threshold are assumed not to be defective and so are labelled as such for the aforementioned image construction. 10733 windows were extracted from the 36 training images and 4617 were extracted from the 18 test images.

A number of features are extracted from each window and these features are used to train the Random Forest. Once the Random Forest has been trained, a dataset of test images are used to analyse the performance and create a Free-response Receiver Operating Characteristic (FROC) curve

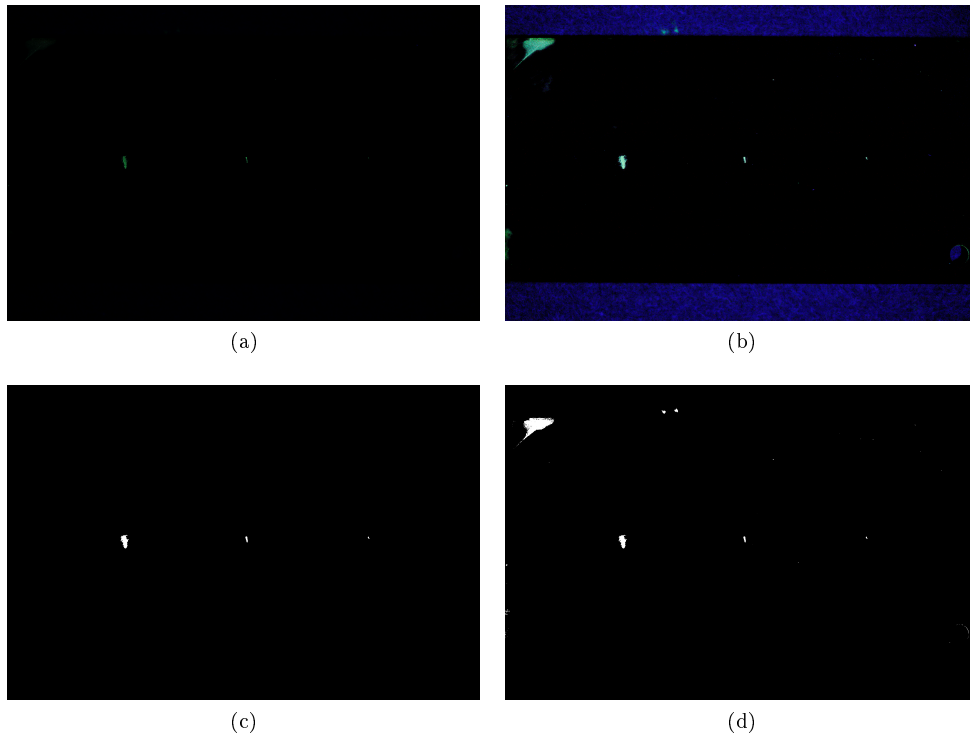


Figure 2: Various images of POD Calibration Plate (W1351). (a) - (d) show original image, post-processed image, image mask (demonstrating ground truth), and binary threshold image respectively.

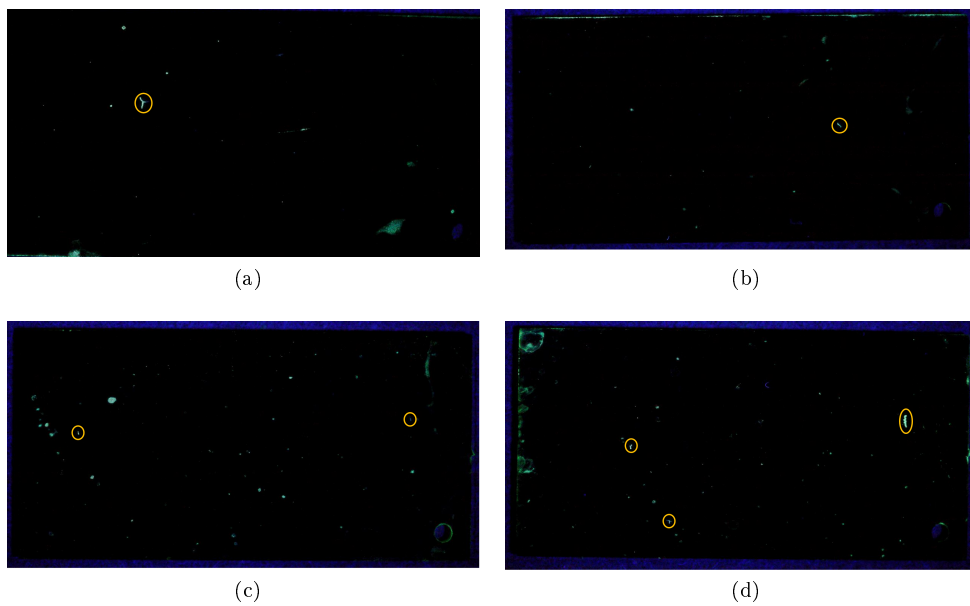


Figure 3: Example images to demonstrate levels of background. Defects circled.

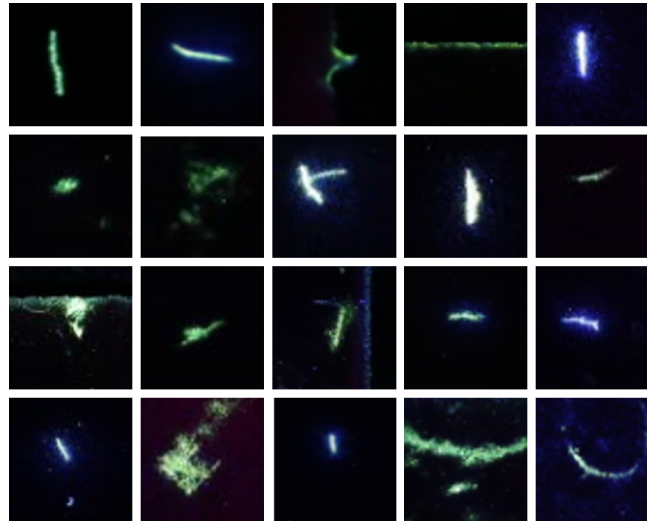


Figure 4: Example windows

of true positive rate against the number of false positives per image.

3.1. Feature Extraction

Once the windows containing the regions of interest had been determined, a number of features were extracted for each region. The reasoning behind the selection of each feature is set out under the headings below.

Peak brightness. This function finds the maximum value in the window. This feature is extracted for a greyscale window and also for each of the red, green and blue levels creating four features in total. This feature was selected as often defects appear brighter than false indications.

Haar features. Haar features are digital image features which consider adjacent rectangular regions. Haar features can be quickly calculated which gives them an advantage over alternatives. A total of five Haar features were extracted for this application. Rectangular Haar feature split the data into rectangles, sum the values in each rectangle and then find the difference between rectangles. For example, a two rectangle vertical Haar (Figure 5a) would split the data into two rectangles vertically and then subtract the data on the left from the data on the right. The same is done horizontally (Figure 5b). A three rectangle vertical Haar (Figure 5c) would extract the left most quarter of the data and the right most quarter of the data from the central half. Again, this is repeated horizontally (Figure 5d). The final Haar feature is a four rectangle (Figure 5e) whereby the top left quarter and bottom right quarter are extracted from the summation of the top right quarter and the bottom left.

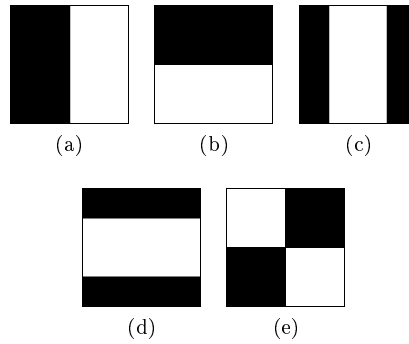


Figure 5: Regions used for calculation of Haar-like features. (a) and (b) show two rectangle vertical and horizontal respectively, (c) and (d) show three rectangle vertical and horizontal respectively, and (e) shows four rectangle.

Area of bright regions. The number of pixels with a brightness value greater than 1000 is summed for each greyscale window. The same is repeated for each of the red, green and blue levels. This feature has been used as often false indications are either very small specks of penetrant or large regions due to poor wash-off.

Average of bright pixels . Similarly, the average value of all the pixels within a window with a brightness value greater than 1000 is also found for greyscale, red, green and blue windows. As with peak brightness, this feature has been introduced as defects can often appear brighter than false indications.

Standard deviation of bright pixels. The standard deviation of pixels with a brightness value greater than 1000 is found for greyscale, red, green and blue windows. Standard deviation has been included to investigate whether penetrant associated with defects is more consistent than that associated with false indications.

Once the above features have been extracted they are combined to form an $N \times 21$ matrix (N being the number of windows, 21 being the total number of features) which will be used to train the Random Forest. The image annotations discussed above are used to give each window a label indicating if it contains a defect or not.

3.2. Random Forest Overview

The Random Forest algorithm works by building a number of decision trees to create a forest. A decision tree is built by segmenting the data according to a number of decisions. For each tree, two features are selected from a random subset of the total number of features. Similarly, bootstrap

sampling is performed to create a subset of the training examples (i.e. windows). For example, say there were 100 training examples each labelled from 1 -100, bootstrap sampling would randomly select a number between 1-100 100 times using a uniform distribution.

Once the two features and the data subset have been selected the decision tree can be trained. Considering the example given in Figure 6, where the two features *peak brightness* (P) and *area of brightness greater than 1000* (A) have been selected, all the data is considered at the first node (Figure 6a). The data is then split as best it can using one line (in this example, the line must be one dimensional). To find the position of this line, one of the two features is selected at random, and a number of random positions are tried (the number of random positions tried is the number of iterations referred to in section 3.3). The position which provides the maximum information gain is selected as the location for the line. For the example in Figure 6, the location is $P = 1000$. Therefore all windows with a peak brightness less than 1000 are passed to the second node (Figure 6b) where they are given a red classification, i.e. that window is not defective. All other windows are passed to the third node where the process is repeated. The complete decision tree is shown in Figure 6c. To continue training the Random Forest, the process of selecting features and training examples and then training a decision tree is repeated until all the trees in the forest are grown.

To use the Random Forest to make a prediction, an image is considered by the software. Again, a brightness threshold is performed to extract regions of interest. As with the training process, features of these regions are extracted and then for each region, the features are passed down the decision trees in the random forest and each tree votes for whether it thinks the region is defective or not. For a random forest of 100 trees, if 25 trees voted that a region was defective and 75 trees voted the region to be defect free, the system would output that it was 75% confident the region is defect free.

3.3. Random Forest Training

For a Random Forest to be trained, there are a standard number of variables which must be selected by the operator [12, 20]. Once the variables have been specified by the operator, the system begins to train each decision tree using the data split methods detailed below. The Random Forest was trained in MATLAB using open-source code made available by Andrej Karpathy [20]. Trials were done altering the following variables to find the optimum performance.

- Feature subset size - proportion of the total number of features which are available for random selection at each node

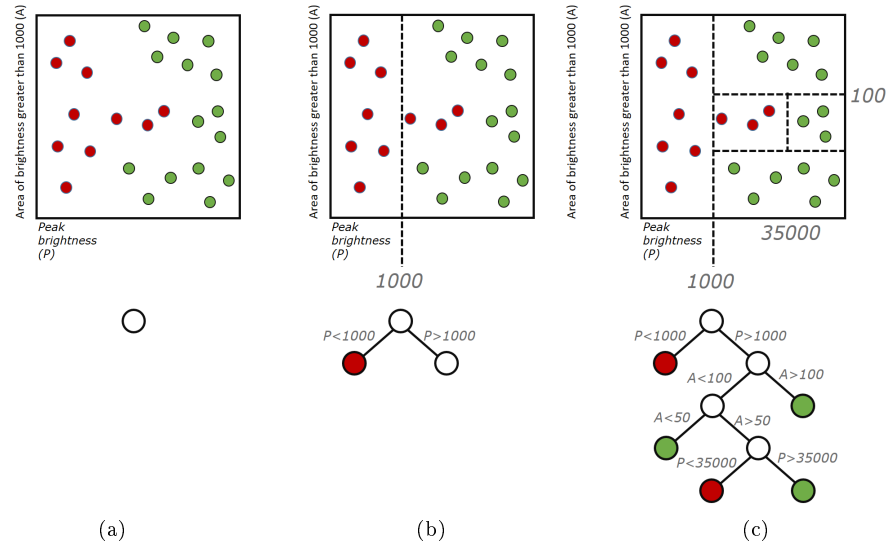


Figure 6: Building up of a decision tree from the first node, (a), to nodes 2 and 3, (b), to the full decision tree (c). Red nodes shows the prediction of a false indication and green nodes show the prediction of defects.

- Data subset size - proportion of the total number of windows which are available for training at each node
- Number of iterations - the number of iterations performed to find the location of the threshold line at each node
- Available data split methods - four methods are available in order to split the data at each node [20]. This variable determines which of the four methods are available for selection at each node. The four methods are;

1. **Decision stumps.** Select a feature at random. Run 5 iterations considering only that feature. For each iteration, select a threshold at random from within the current range of that feature. Classify data based on a threshold value for a particular feature. Evaluate performance using Shannon entropy and information gain. Iterate through total number of random splits, changing the feature every 5 iterations. The feature/threshold combination which provides the maximum information gain is selected.
2. **2D linear decision learner.** Select two features, x_1 and x_2 , and three constants, ω_0 , ω_1 and ω_2 , at random. $y_{pred} = \omega_0 + \omega_1 x_1 + \omega_2 x_2$. If y_{pred} is less than 0, predict 1, else predict 0. For each iteration, randomly select another two features and three constants. The feature/constant combination which provides the maximum information gain is selected.

3. **Conic section learner.** Select two features, x_1 and x_2 , and six constants at random, ω_0 , ω_1 , ω_2 , ω_3 , ω_4 and ω_5 . Create a new matrix mv , where $mv = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \omega_3 x_1 x_2 + \omega_4 x_1^2 + \omega_5 x_2^2$. Select two random threshold values t_1 and t_2 . Another random variable is selected between 0 and 1. If this number is less than 0.5, the prediction is 1 if $mv < t_2$, else the prediction is 1 if $mv > t_1$ and $mv < t_2$. For each iteration, randomly select another two features, six constants and two threshold values. The feature/constant/threshold selection which produces the maximum information gain is selected.

4. **Distance learner.** Create a training example type matrix with one random integer value selected for each feature. This point is then plotted in m -dimensional space (where m is the number of features) along with all data points. The Euclidean distances between all points and the random integer point are found. A threshold value is then selected within the range of Euclidean distances, predicting 1 for values above then threshold and 0 for those below. The feature/threshold combination with the maximum information gain is selected.

- Data split commit first - this determines whether the system randomly selects a data split method and trains the node using that method for all iterations or whether the system performs the iterations for each of the available data split methods and then selects the method which provides the maximum information gain.
- Number of trees
- Depth of trees - how many decisions each data point will pass through when travelling down a particular decision tree.

4. Results

Tests were done using 36 images to train, 18 images to test. Each of the variables listed in Section 3.3 was changed independently with a goal of establishing the performance characteristics. FROC plots [13, 24] demonstrating the effects of changing each variable are shown in Figures 7 to 13. An FROC curve is a modification of the ROC curve [34, 18] which is applicable for when there could be more than one defect within an image. For each image, the regions of interest, or marks, are each given a rating according to the probability of the region being defective. The FROC curve plots the true positive fraction (or lesion localisation fraction, LLF) against the number of false positives per image (or non-lesion localisation fraction, NLF). The plot can extend indefinitely in the x-direction but is

limited to between 0 and 1 in the y-direction. Each successive point on an FROC curve is produced by varying the probability threshold of each mark rating from 0 to 1. The ideal position on the FROC curve is at (0,1), where all true positives have been detected with no false positives.

Figure 7 to 13 also show the data point from a human inspector. The Rolls-Royce specification for the Training and Approval of NDT Personnel within Rolls-Royce UK Gas Turbine Operations [3] specifies that in order for an inspector to be able to evaluate parts for acceptance or rejection, they must have the minimum of a Level 2 qualification. The Level 2 qualification is defined by the European standard for personal certification and approval for NDT in the aerospace industry [2]. A Level 2 dye penetrant inspector was therefore used to compare results. The Level 2 inspector was shown the images seen by the software. The inspector was given the ability to brighten the image should they wish, and then were asked to record regions they believed to be defects. The Level 2 inspector correctly identified 74% of the defects whilst making one false call every five images.

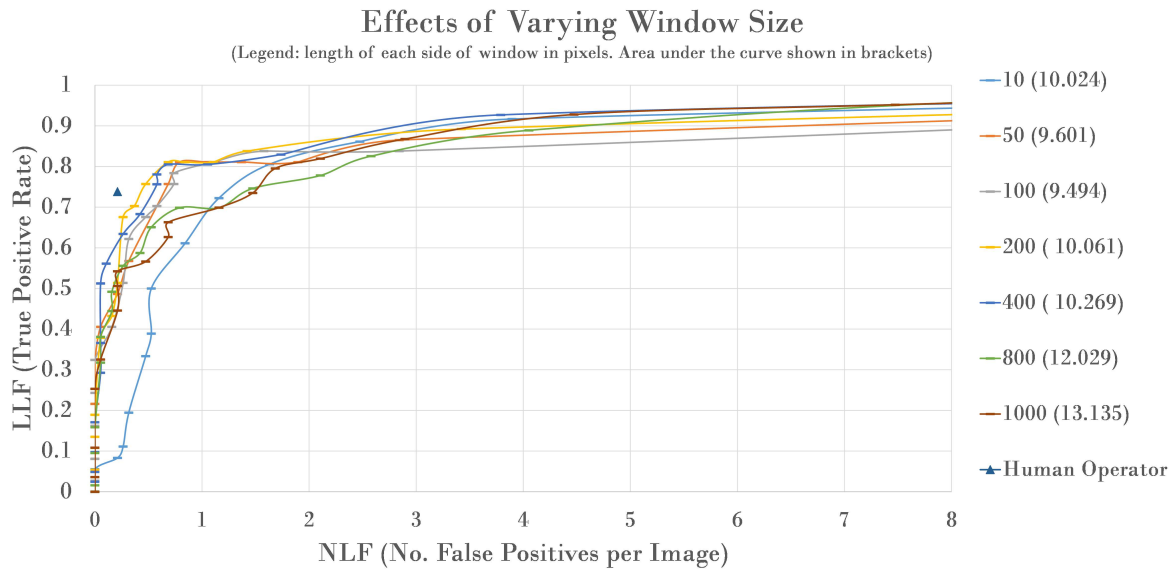


Figure 7: FROC curve varying window size (area under each curve is shown in brackets)

Figure 7 shows the FROC curve demonstrating the effect of varying the window size. The legend shows the length of each side of the window, in pixels, for example 50 would be a 50×50 window. The number in brackets shows the area underneath each curve. The largest defect in the dataset is 3.4 mm and the indication provided by this defect is 170 pixels at its longest point. A window size of 400 was decided on as the optimum size. A size of 400 ensures the window encompasses even the largest defects whilst ensuring a reasonable true positive rate at a low false positive rate.

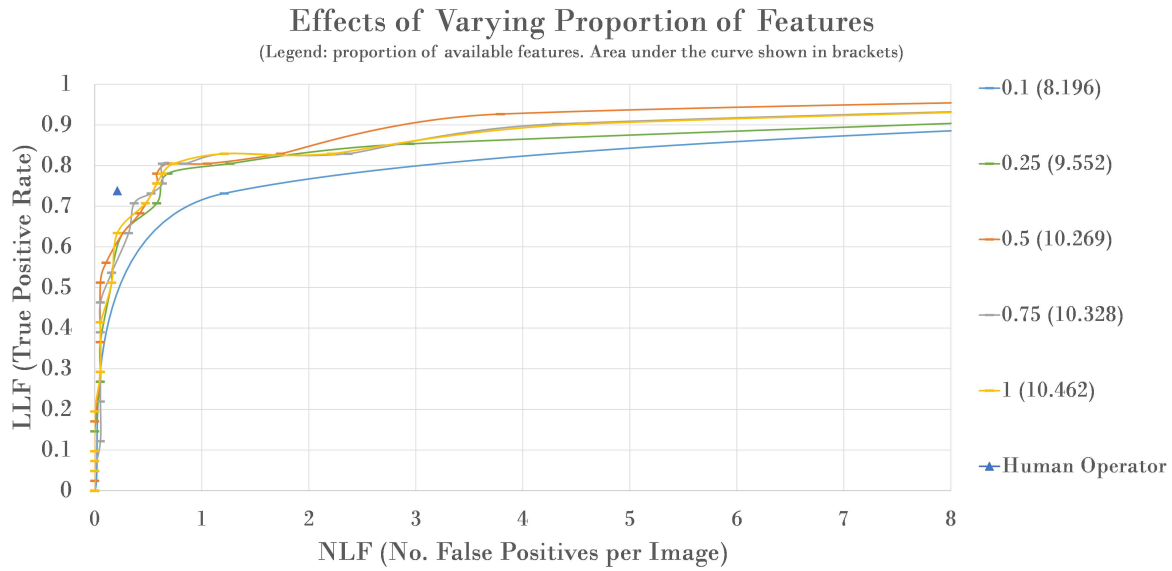


Figure 8: FROC curve varying proportion of features available for selection at each node

Figure 8 shows the effects of varying the proportion of features available for selection for each tree. It can be seen that increasing the proportion of features available for selection at each node increases the area under the curve, although it should be noted the increase in performance is minimal. A feature proportion of 1 was therefore selected.

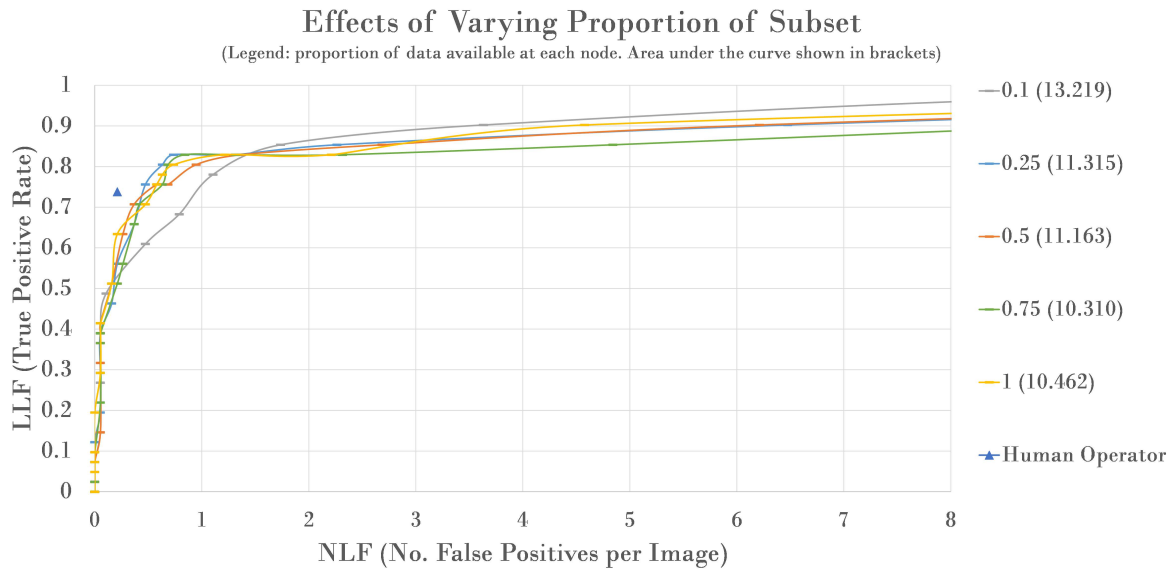


Figure 9: FROC curve varying proportion of data subset available for training at each node

Figure 9 shows the effect of varying the proportion of training data available for selection for each

tree. It can be seen that reducing the proportion available increases the area under the curve. Having a proportion of 0.1 available detects fewer true positives at a false positive rate of less than 1 and therefore it was decided to use a data subset proportion of 0.25.

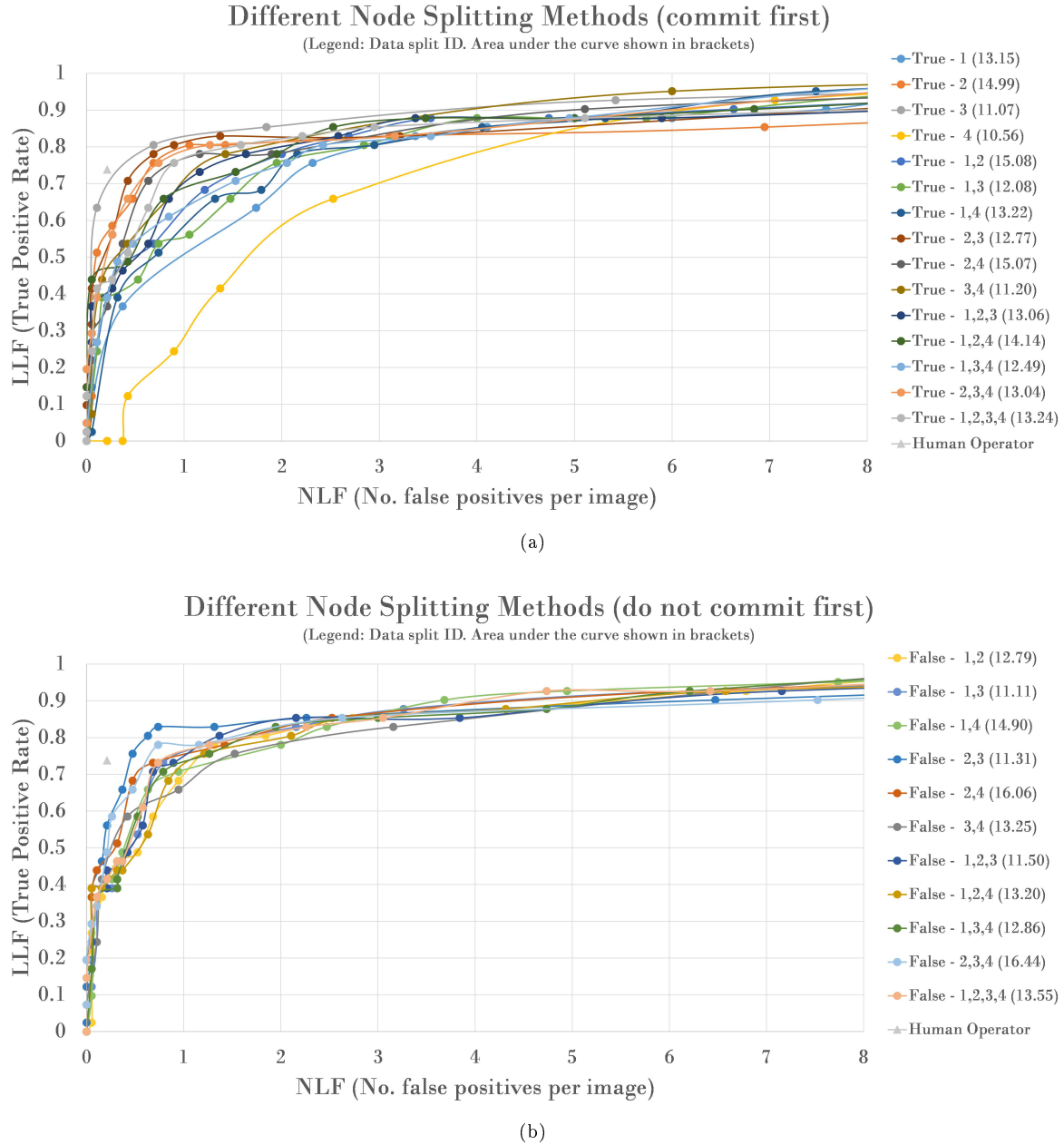


Figure 10: FROC curve varying data split methods

Figure 10 shows the effects of the data split methods detailed in Section 3.3 (the numbering in the legend is the same as that in Section 3.3 with 1 referring to decision stumps, 2 being 2D linear decision

learners, 3 conic section learners and 4 is distance learners). Figure 10a shows all of the combinations of classifiers whereby the system randomly selects one of the available classifiers and trains on that. Figure 10b shows the combinations of classifier ID whereby all available classifiers are run and the option providing the maximum information gain is selected. The greatest area under the curve was given by 'False - 2, 3, 4', i.e. for each node, the data was split using 2D decision learners, conic section learners and distance learners and then the option offering the maximum information gain was selected. This option was used as the optimum setting.

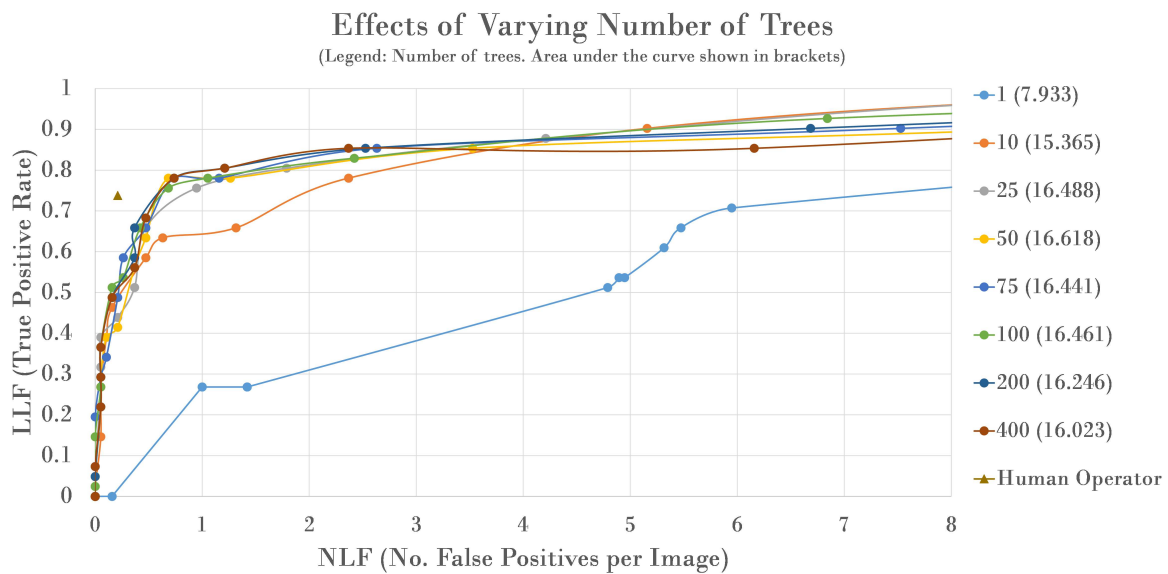


Figure 11: FROC curve varying number of trees

Figure 11 shows the effect of varying the number of trees within the Random Forest. The more trees within the forest, the longer the computation time to both train and classify. (Using a single core of an Intel Xeon E5-2667 CPU running at 3.20GHz, it takes around 20 seconds to train each tree in the forest with a MATLAB implementation of the technique when each tree had a maximum depth of 17.) It can be seen from Figure 11 that provided there are more than around 10 trees in the forest, the effect of varying the number of trees is minimal. The best performance was found to happen when there are 25 trees trained, however, in order to limit susceptibility to overfitting, 100 trees was selected as the optimum.

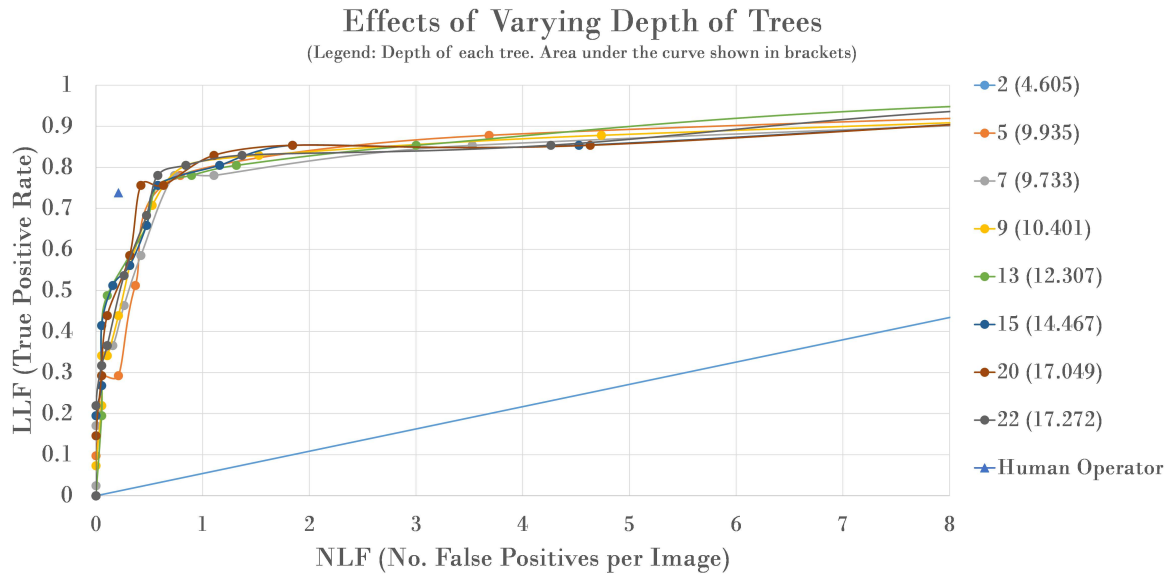


Figure 12: FROC curve varying tree depth

Figure 12 shows the effect of varying the depth of tree. (Considering Figure 6, Figure 6a has a depth of 1, Figure 6b has a depth of 2, Figure 6c has a depth of 5.) It can be seen that increasing the depth of trees in the forest increases the area under the curve however this comes with an increase computation time. For example, training a Random Forest with trees at a depth of 22 comes with a computation time four times longer than that with a depth of 20. (Using a single core of an Intel Xeon E5-2667 CPU running at 3.20GHz, it takes around 2 hours to train 100 trees with a MATLAB implementation of the technique when each tree had a maximum depth of 20. When the depth was raised to 22, the computation time to train the forest increased to 8 hours.) Setting the depth at 20 still offers an acceptable computation time and as it also offers the largest area under the curve, this was selected. However it is noted that with the increasing speed of computers, run time considerations become less important.

Variable	Value
Window size	400
Feature subset size	1
Data subset size	0.25
Available data split methods	2D decision learners, conic section learners, distance learners
Data split commit first	False
No. trees	100
Depth of trees	20
No. iterations	30

Table 2: Optimum settings

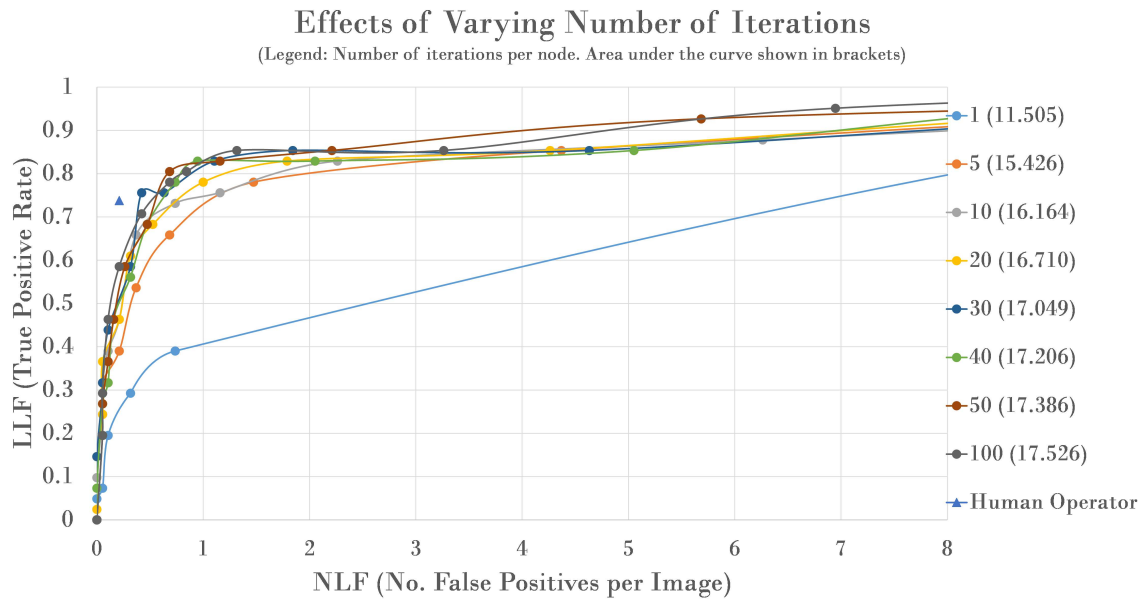


Figure 13: FROC curve varying number of iterations

The final parameter which needs to be tuned is the number of iterations performed at each node. This is demonstrated in Figure 13. Similar to the depth of trees, both performance and computation time increase as the number of iterations is increased. It is therefore necessary to find the point at which there is minimal increase in performance. This was found to occur at around 30 so it was decided 30 was the optimum number of iterations to perform.

The settings and results of the optimum performance are shown in Table 2 and Figure 14 respectively.

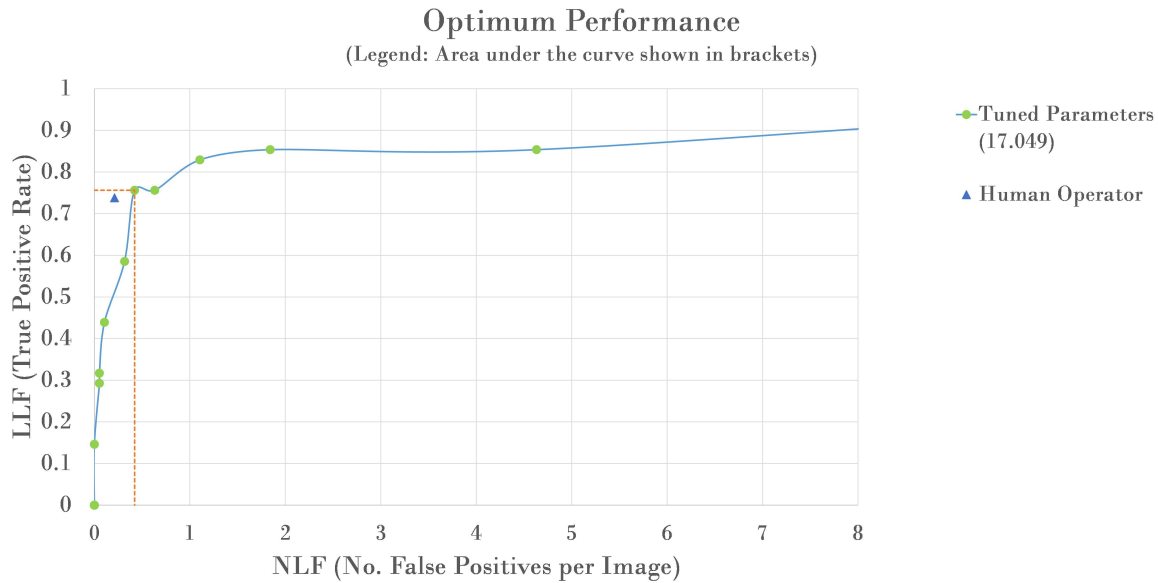


Figure 14: FROC Curve of optimum results achieved to date

5. Discussion and Conclusions

Figure 14 shows the optimum FROC curve produced to date using this method. The results shown in Figure 14 demonstrate the system's capability to distinguish between penetrant associated with defects from that linked to poor processing and geometrical indications. The system was able to correctly identify around 76% of defects whilst identifying less than one false call every two images; with the threshold value at 0.35 (see dotted lines in Figure 14), the True Positive Rate (Y-axis) is 0.76 and the number of False Positives per Image (X-axis) is 0.42, see Figure 14. It can be seen that this is comparable to the Level 2 inspector who correctly identified 74% of the defects whilst making one false call every five images.

It can be concluded that the Random Forest method demonstrates good capability at distinguishing between penetrant associated with defects from background noise even with a very small training set. Whilst the number of false positives is twice that of the human inspector, the system has correctly identified 76% of defects, demonstrating capability comparable to that of the human inspector. It is believed that the capability of the system can be improved further by training with more data sets and investigating additional features. Ongoing work will include the investigation of these.

Other future work will also involve considering the inspection of real components rather than test pieces. The first application is likely to be a component with a geometrically simple shape. It is hoped test piece images obtained to date will be able to be used as training examples for the next component

application.

More broadly, the strong performance of the random forest in the specific application considered here, with the challenging issues surrounding background noise and restricted numbers of data sets, suggest that it could well be suitable for other NDE applications, including radiographic and ultrasonic imaging. Within FPI applications, since the RF itself has clearly demonstrated good potential on simple flat plates, it may be that it still works on more complex components without having to retrain the system. Future work will therefore involve considering the inspection of real components rather than test pieces. The first application is likely to be a component with a geometrically simple shape in order to assess this.

Acknowledgements

This work was supported by the Engineering & Physical Sciences Research Council Grant number EP/L015587/1, EP/M020207/1 and by Rolls-Royce plc. The authors wish to thank Professor Timothy Cootes (University of Manchester) for his advice on the use, and implementation of the Random Forest algorithm.

-
- [1] AMS2644F - Inspection Material, Penetrant. AMS Non Destructive Methods and Processes Committee, SAE International.
 - [2] Aerospace Series - Qualification and Approval of Personnel for Non-Destructive Testing. Technical Report BS EN 4179:2017, British Standards Institution, 2017.
 - [3] Training and Approval of Non-Destructive Testing Personnel within Rolls-Royce UK Gas Turbine Operations. Technical Report RPS915, Rolls-Royce plc, 2017.
 - [4] Machine Learning: The Power and Promise of Computers that Learn by Example. *The Royal Society Publication. (2017)*, 2017.
 - [5] Klaus Abend. Fully Automated Dye-Penetrant Inspection of Automotive Parts. *Computer Standards & Interfaces*, 1999.
 - [6] T.L. Adair, D.H. Wehener, M.G. Kindrew, H.I. Winter, and B. MacCracken. Automated Fluorescent Penetrant Inspection (FPI) System is Triple A. *1998 IEEE AUTOTESTCON Proceedings*.

IEEE Systems Readiness Technology Conference. Test Technology for the 21st Century (Cat. No.98CH36179), (724), 1998.

- [7] Cesar Augusto Correa Alba. *Image Acquisition and Processing in an Attempt to Automate the Fluorescent Penetrant Inspection*. PhD thesis, École de Technologie Supérieure Université du Québec, 2011.
- [8] Francis Alexander. Machine Learning [Guest editor's introduction]. *Computing in Science & Engineering*, 15(5):9–11, Sept - Oct 2013.
- [9] Mariana Belgiu and Lucian Dragut. Random Forest in Remote Sensing: A Review of Applications and Future Directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, pages 24–31, 02 2016.
- [10] Gerard Biau and Erwan Scornet. A Random Forest Guided Tour. *TEST*, 25(2):197–227, 06 2016.
- [11] L Brasche, R Lopez, and B Larson. A Study of Drying and Cleaning Methods used in Preparation for Fluorescent Penetrant Inspection. In *AIP Conference Proceedings*, volume 657, 2003.
- [12] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 08856125.
- [13] Dev Chakraborty. Basics of ROC and FROC. 2005. URL <http://www.devchakraborty.com/index.php>.
- [14] Jonathan Cheung Wai Chan, Pieter Beckers, Toon Spanhove, and Jeroen Vanden Borre. An Evaluation of Ensemble Classifiers for Mapping Natura 2000 Heathland in Belgium using Spaceborne Angular Hyperspectral (CHRIS/Proba) Imagery. *International Journal of Applied Earth Observation and Geoinformation*, 08 2012.
- [15] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20:273–297, 1995.
- [16] J S Cramer. The Origins of Logistic Regression. Technical report, Tinbergen Institute, 2002.
- [17] K. Daneshvar and B. Dogan. Application of Quantum Dots as a Fluorescent Penetrant for Weld Crack Detection. *Materials at High Temperatures*, 27(3):179–182, 2010.
- [18] DJ Hand. Evaluating Diagnostic Tests: The Area Under the ROC Curve and the Balance of Errors. *Statistics in Medicine*, 29(14):1502–1510, 2010.

-
- [19] International Atomic Energy Agency (IAEA, editor. *Non-destructive Testing: A Guidebook for Industrial Management and Quality Control Personnel (IAEA-TCS-9)*. International Atomic Energy Agency, 1999.
- [20] Andrej Karpathy. Random Forest for Matlab. GitHub, 2012. URL <https://github.com/karpathy/Random-Forest-Matlab>.
- [21] B Larson. Study of the Factors Affecting the Sensitivity of Liquid Penetrant Inspections: Review of Literature Published from 1970 to 1998. Technical report, 2002.
- [22] David J. Livingstone, editor. *Artificial Neural Networks Methods and Applications*. Humana Press, 2008. ISBN 978-1-58829-718-1.
- [23] Lufthansa Technik AG. Engine Overhauls: The Search for Material Damage. URL <https://www.lufthansa-technik.com/engine-overhauls>.
- [24] Harold Miller. The FROC Curve: A Representation of the Observer's Performance for the Method of Free Response. *The Journal of the Acoustical Society of America*, 46(1473), 1969.
- [25] National Transportation Safety Board. Aircraft Accident Report: United Airlines Flight 232, McDonnell Douglas DC-10-10. Technical report, 1990.
- [26] National Transportation Safety Board. Aircraft Accident Report: Uncontained Engine Failure/Fire Valujet Airlines Flight 597 Douglas DC-9-32. Technical report, 1996.
- [27] National Transportation Safety Board. Aircraft Accident Report: Uncontained Engine Failure Delta Air Lines Flight 1288 McDonnell Douglas MD-88. Technical report, 1998.
- [28] Vili Podgorelec and Milan Zorman. *Decision Trees*, pages 827–845. Springer, New York, 2012.
- [29] Rolls-Royce plc. RRP 58003 Aerospace Process Specification: Fluorescent Penetrant Inspection (FPI), 2016.
- [30] Rolls-Royce plc. RRP 58000 Technical Control of Non-Destructive Testing, 2017.
- [31] S.A. Sattar and C.V. Sundt. Gas Turbine Engine Disk Cyclic Life Prediction. *Journal of Aircraft*, 12(4):360, 1975.
- [32] Mehmet Simsira and Alpay Ankarab. Comparison of Two Non-Destructive Inspection Techniques on the Basis of Sensitivity and Reliability. *Materials & Design*, 28(5):1433–1439, 2007.

-
- [33] Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer Science+Business Media, LLC, 2008.
- [34] John Swets. *Signal Detection Theory and ROC Analysis in Psychology and Diagnostics: Collected Papers*. Psychology Press, 2014.
- [35] Yonghong Tang, Aiqun Niu, William G Wee, Chia Yung Han, and Computer Science. An Automated Inspection System for Detecting Metal Surface Cracks from Fluorescent Penetrant Images. *Machine Vision Applications in Industrial Inspection III*, 1995.
- [36] J. Zheng, W. F. Xie, M. Viens, L. Birglen, and I. Mantegh. Design of Advanced Automatic Inspection System for Turbine Blade FPI Analysis. In *AIP Conference Proceedings*, volume 612, 2013. ISBN 9780735411296.