Automatic Crack Opening Displacement Measurement Using Image Processing

Robert Sloan, Ekaterina Tsypkaikina, Lancelot Chen
School of Computer Science and Engineering
The University of New South Wales
Sydney, Australia

***Abstract*—Crack Opening Displacement (COD) measurement is an important step for determining material properties. However, manual measurement is time consuming and labour intensive. In this project, measurement of CODs using image processing techniques is investigated. Preprocessing techniques are trialled and an algorithm for calculating CODs from binarised images was proposed and tested. Further work is needed to fully verify the calculation algorithm and to find more suitable and robust preprocessing methods.**

***Index Terms*—crack opening displacement, image processing**

I. Introduction

Assessment of engineering materials requires determining the materials properties using available measurements. In understanding fracture characteristics and cracking behaviour, the crack opening displacement (COD) is an important metric used for calculating various properties. The critical COD value itself has been used extensively in the prediction of the onset of crack initiation. Hence COD is an extremely useful metric, in some cases being the only measurable parameter in fracture tests. [1, 2]

COD is usually generated by using mathematical models and finite element methods. [1]. It is also directly measured using scanning electron microscopes (SEM). [3]. These methods are time consuming and labour-intensive. In this paper the use of image processing techniques to automatically measure COD was investigated. Various preprocessing techniques were trialled on SEM images, and an efficient algorithm for measuring COD given an appropriately processed image was developed.

The work presented provides a proof of concept of the process, but further work will provide more functionality to researchers using this application.

II. Crack Opening Displacement

The crack opening displacement can be visualised as the width of the crack at a given point along the length of the crack, measured at 90 degrees to the axis of the crack. Since the crack may not always travel in the same direction, as in the case of a Vickers test, the direction of the crack must be considered at the point of measurement. Fig. 1 shows the necessary adjustments to be made when measuring in relation to a global crack plane. $2\delta_n$ is the relevant COD. [3]

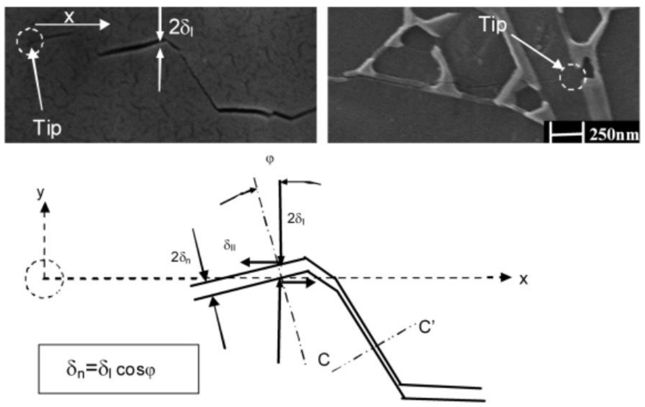


Fig. 1. Crack diagram [3]

III. Current Research

While crack identification has received much attention in modern applications [4, 5], algorithms for crack quantification are sparser, and currently undeveloped for the quantification of COD.

Previous methods of crack detection have used a combination of filters and thresholds to preprocess the images. Commonly used filters and thresholds included the median filter, Sobel filter, Laplacian of Gaussian filter [6], Canny edge detection, and Otsu thresholding [4].

One effective crack detection method uses a ”percolation” technique, where the crack is determined through the flow of similar neighbouring pixels [5]. This method may also be applicable to the cracks in our dataset. Convolutional neural networks have also proven to provide accurate crack detection [7], but due to the limited sample size of the dataset, more traditional approaches for the crack detection algorithm were used.

In one application, image processing techniques were used in the quantification of cracks in chromium electrodeposits [6], with the aim of providing an objective measure of the crack area. The traditional approach has been to manually and subjectively describe the crack area, but the use of image processing allowed a consistent and objective method. Vidal et al. tried various preprocessing techniques to enhance cracks in their images, including Canny edge detection, Prewitt operators and Laplacian of Gaussian (LoG) methods. They also proposed their own thresholding function for binarisation of the crack [6]. Enhancement of the crack image for binarisation was also an important part of the method utilised in this paper, so techniques used by Vidal et al. were also trialled and evaluated.

IV. Project Scope

The overall aim of this project was to find a method and create an application which would automate the measuring of COD. However, given the limited timeframe and challenges met during the project, it was not possible to create a fully working application. Therefore, our aims were revised to:

- Investigate available data and find an appropriate format
- Investigate image preprocessing techniques including methods for noise removal, edge enhancement and thresholding
- Create an algorithm for calculation of COD
- Provide proof-of-concept of the COD algorithm and overall workflow

Further items that are being developed but have not been complete are:

- GUI allowing manual selection of points to process and processing single and multiple images
- Virtual stitching of images of a single crack to allow querying points on the crack along a global axis

V. Problem Decomposition

As the project scope evolved, the task was naturally broken down into three major components.

1) Handling and parsing input data, including images and scale informations
2) Preprocessing and binarisation of images
3) Calculation of COD from binary images

VI. Data

Datasets used in this task comprised of series of SEM images of cracks in various ceramics. An example of an image is given in Fig. 2. These cracks were produced using a Vickers indenter, and images at different magnifications were taken along the length of the crack. In addition to the images, scale information in the form of pixels per distance was provided, as well as the COD and location of different points in the crack. This information was contained in a spreadsheet.

Scale information was extracted from the spreadsheets where possible and placed in a CSV file, which was parsed by the developed application. Not all images had scale information available. However, since some magnifications and their corresponding distance per pixel values were provided, it is possible to interpolate for the other magnifications.

It is important to note that the data was not taken with automation of COD measurements in mind. Hence, the images lacked global location information which would allow an entire crack to be reconstructed in the application. The overlap between images were also randomly chosen, using visual features which were easy for the researcher to identify and memorise when moving between images. As a result, they are not, and in some cases the overlap between images was difficult to identify.

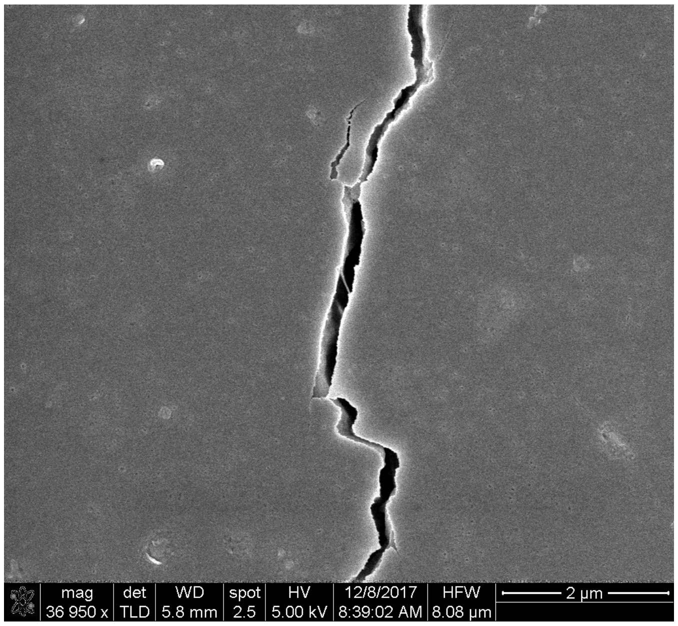


Fig. 2. SEM image of section of crack

The points at which the COD were manually measured were also chosen based on ease of measurement. Since COD is measured at 90 degrees to the crack axis, only points where the crack ran parallel to the direction of travel i.e. along the global axis were considered for measurement. Furthermore, any areas affected by crack bridging were excluded. For evaluation, in order to compare automatic and manual measurements, a manual selection of the points to measure with the algorithm was required, by selecting the points with a graphical user interface (GUI).

Further work will aim to reconstruct an entire crack from a set of images. This will allow the user to specify any point on a global axis along the length of the crack and find the COD, without the use of a GUI. This will provide greater automation of the process. However, to do this, the images must be taken with a large amount of overlap. Ideally, the amount of overlap should be a fixed amount, but if this is not possible, feature detection using SURF can be performed and matching descriptors found. For images of the same scale, a Euclidean transformation matrix can be calculated using the matching descriptors. Assuming no rotation of the specimen or SEM, the translation and hence overlap can be found from the transformation matrix. If the images are ordered, then by taking the beginning of the first image as the origin, the crack can be aligned on the global axis in consecutive images of the same scale. Where the scale changes, the distance of the beginning of the image from the beginning of the first image will need to be manually measured and specified as additional input data. From there the distances of points in following images in the new scale can be calculated.

VII. Preprocessing

The goal of the preprocessing stage was to acquire a binary image that accurately distinguishes the crack from the background, as shown in Fig. 3.

Research on this subject showed several possible approaches to the problem [4–7], including machine learning and traditional image processing. Machine learning and neural networks were not chosen due to the limited data sets available, and because the continuously varying output of the application was considered unsuitable for neural networks. Work performed by Vidal et al. [6] was very relevant to this application, and thus a simiar approach was taken to see how it would perform on the available data.

The adapted algorithm first performed contrast adjustment through histogram equalisation, after which the Laplacian of Gaussian was applied, with an additional Prewitt edge-detection filter in some cases. After this the image was binarised based on a threshold calculated based on the output image histogram [6].

The main issues that were encountered in this process arose due to noise, which was heavily present in most cases. Using too much blurring to remove the noise gave an incorrect outline of the crack, leading to difficulties in detecting the correct crack edges. In addition, the presence of crack bridging effects meant that the crack was not always a single continuous body, and that the main crack could consist of multiple smaller entities. These needed to be distinguished from cracks that were not part of the main crack body.

To address these issues, multiple experiments with parameter tuning were carried out, using additional filters such as Sobel edge-detection, but the noise was almost always present on the resulting image, as well as in some of the darker areas of the image.

Based on additional research [8], a clustering approach was chosen. A two-cluster separation of image pixels based on their intensity, using k-means clustering, proved to be more flexible, and additionally solving the issue of cracks in bridging areas. It has also been noted that a brightness adjustment with the addition of Gaussian blurring improves the result significantly.

This approach has proven to give considerably better results than the previous thresholding method. However, it is still highly affected by noise and the quality of the images; thus, the parameters still require manual adjustment to fit different cases, as the characteristics of the images vary throughout the dataset. Dealing with noise and smaller cracks remains a priority for future work, especially given the sensitivity of the algorithm to the binarised image quality. It is also possible to consider using contextual information to filter out unwanted cracks, as it is known that the main crack elements are grouped together and travelling along a vertical or horizontal axis.

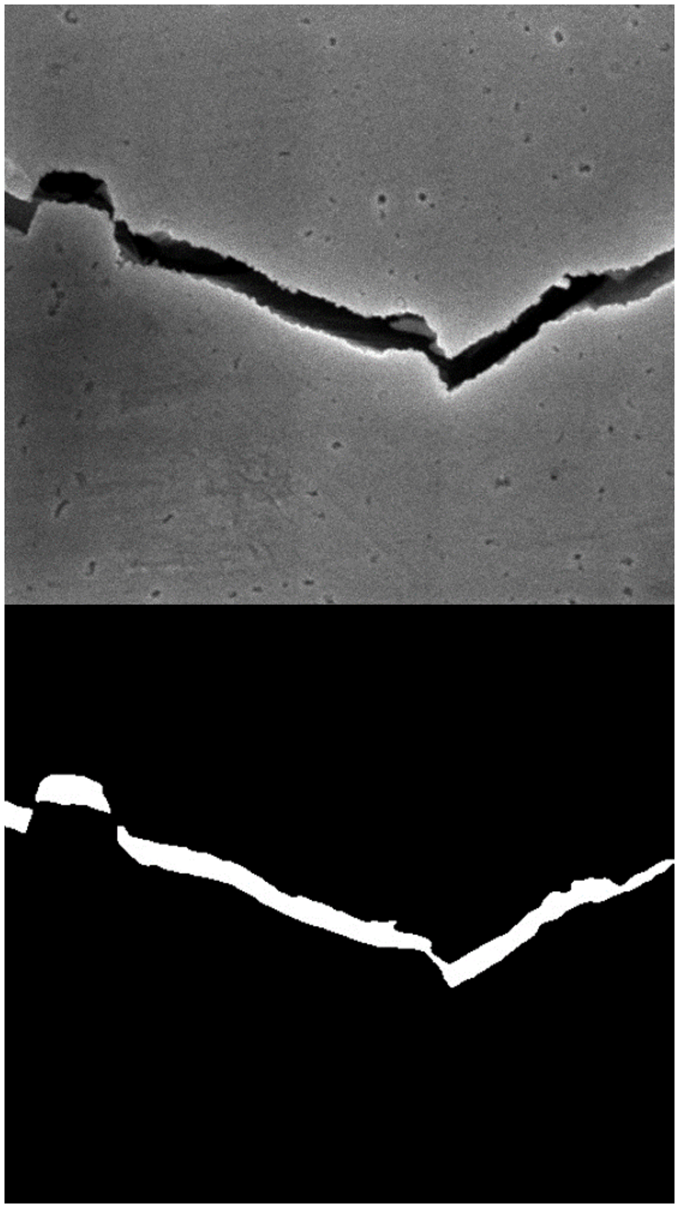


Fig. 3. Example of crack image and its corresponding binary image

VIII. Crack Opening Displacement Calculation Algorithm

A. Mid-point Calculation

If we let the mid-point of the crack be the average of the top and bottom points along the y axis, we do not get the true centre of the crack because the crack curves and is not aligned with a single axis. Finding centre points like this is not rotation invariant and leaves large gaps in the line.

An acceptable centre line for the cracks can be found by eroding the edges of the cracks until they converge. However, simple erosion algorithms remove converging points, so we wrote an algorithm to retain the converging points. The algorithm iteratively processes each layer of each binarised crack until only the converging centre points remain.

An initial pass of all pixels in the binarised image finds the crack edges and adds each one to a queue for processing. And then, iteratively until all the binary points have been processed:

- The points are checked for convergence (whether deleting it will cause a separation between two parts of the crack).

If it converges, then it is saved as a converging point, otherwise it is considered an edge point
- Then, the next layer (the neighbours of each non-converging edge point) is added for processing, and the current non-converging edge points are removed

This results in a collection of rotation-invariant centre points for each of the cracks in the image, as seen in Fig. 4.

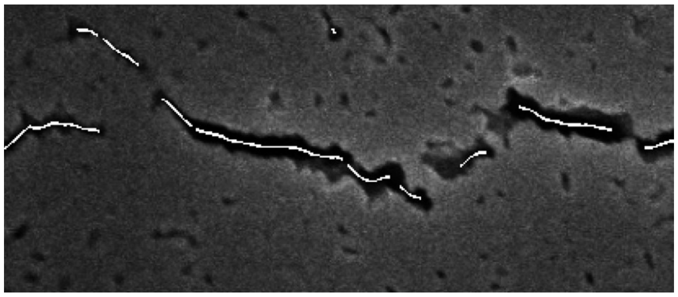


Fig. 4. Centre points in a crack derived from the erosion algorithm

B. Crack Opening Displacement Calculation

After finding the centre line of the crack, we approximate a local gradient for points along the line by averaging the gradients between a window of surrounding points. As the crack may move in any direction, and the gradient may therefore tend towards infinity, the gradient is implemented as a $(\Delta x, \Delta y)$ pair. From these points in the crack with their corresponding local gradients, we can calculate the COD information for each point.

From a given point, the algorithm steps along the perpendicular gradient until it finds the top of the crack, and then steps along the negative gradient until it finds the bottom. The basic point-gradient form of a line is used to calculate the step towards the top/bottom of the crack; if we take $y = m(x - x_1) + y_1$, where m is the perpendicular gradient $-\Delta x/\Delta y$, and (x_1, y_1) is a point in the crack, then we can step along the line by incrementing or decrementing the x position by 1. However, because we are working with discrete x and y values, when $m > 1$, y may step by more than 1 pixel each step, and may therefore overstep the edge and return an inaccurately distant point, or it may step over a gap into the bounds of another crack.

So,

- when $m1$ we step along the x axis, letting $y = m(x - x_1) + y_1$,
- when $m > 1$ we step along the y axis, letting $x = \frac{y-y_1}{m} + x_1$

Additionally,

- when $\Delta x = 0$, we use $y = y_1$ (to minimise computation), and
- when $\Delta y = 0$, we use $x = x_1$ (to avoid dividing by zero).

For each step along the gradient we check the value at $B(x, y)$. If this value is 0, then we have stepped off the crack and can take the previous (x, y) point as the top or bottom of the crack. From these top and bottom points we can simply take the Euclidean distance to get the COD. Fig. 5 shows the perpendicular gradients calculated along the length of the crack.

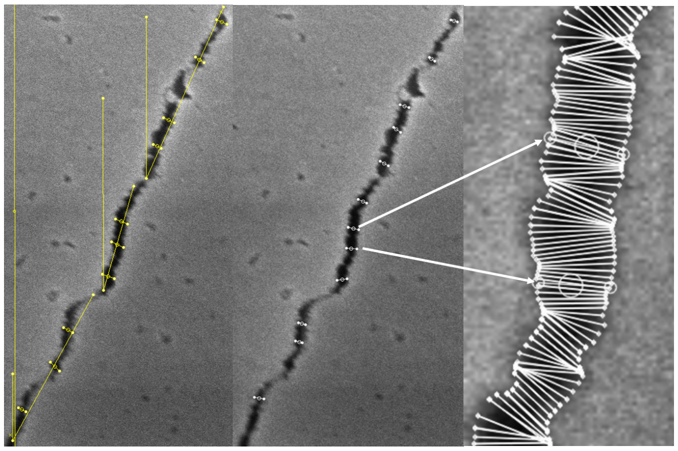


Fig. 5. Perpendicular gradients calculated along length of crack

C. Complete Calculations

The processing step takes a binarised image of the cracks, and returns the COD measurement, gradient, top, mid-point and bottom for each centre point found in each crack in the image. This preprocessing step can be computationally expensive; on our 1024x943 images we found that average calculation was approximately 5 seconds. Here we calculate the COD for every point along the crack, but this full preprocessing step could be exchanged for local real-time calculation if only a selection of points were needed, and the dataset was extensively large. Fig. 6 shows an example of the numerical output that is produced by the calculations.

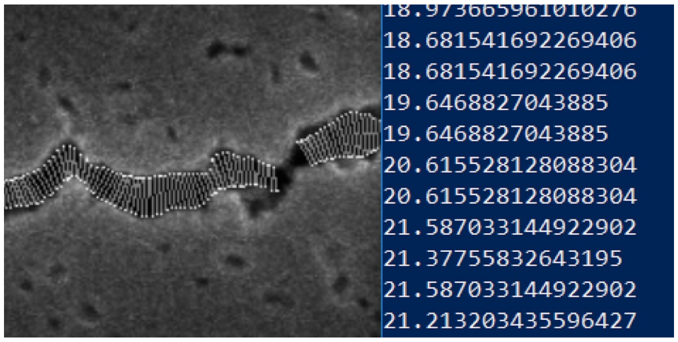
We also upscaled our images as part of the preprocessing step to increase the samples taken and give smoother results. This scale can be adjusted according to the computational complexity of the image.

Many of the algorithms we implemented can be optimised to be several times faster through implementing native C calls, leveraging parallel processing or improving the algorithms. Most of our unoptimized algorithm checks values pixel for pixel in Python causing it to be slower than necessary.

IX. Results

In evaluating our algorithms, we relied heavily on visual inspection. Calculations for given, uncalculated points were found by approximating the gradient based on surrounding measurements within a given window, and those points were visualized in our testing framework. With more time, we would like to integrate the entire dataset to measure our results quantitatively and see which areas of our method needs improving.

A direct comparison of results from the recorded dataset and our recorded results for the same points is show in Fig.



```
18.973665961010276
18.681541692269406
18.681541692269406
19.6468827043885
19.6468827043885
20.615528128088304
20.615528128088304
21.587033144922902
21.37755832643195
21.587033144922902
21.213203435596427
```

Fig. 6. Calculated COD at each gradient

7, where human recorded displacements are in blue, and computer recorded in orange. There is clearly a strong correlation between points; we believe the error in measurements may be from the calculated threshold only taking the interior of the crack, where the human recorded points included the sides of the crack.

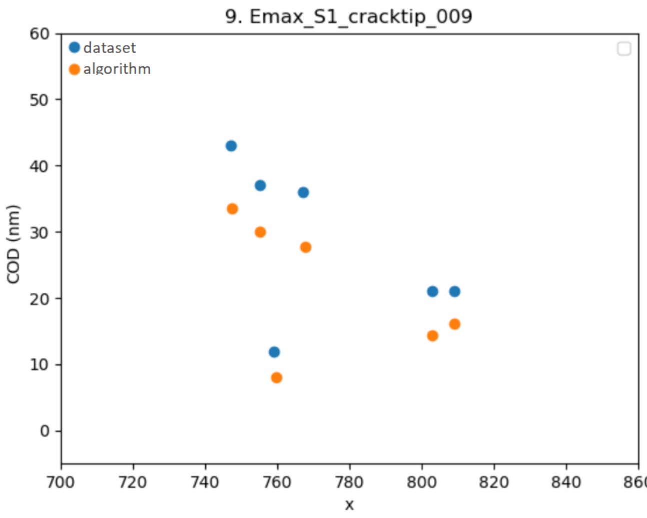


Fig. 7. Results of the algorithm compared with manual measurements

Varying the thresholding also led to changes in the results. Fig. 8 shows the results of the algorithm with a good threshold which produces accurate binarisation, while Fig. 9 demonstrates poor thresholding leading to larger errors in measurements. It is clear the quality of the results is highly dependent on the preprocessing steps. Thus, because of our current preprocessing method, the results are highly affected by the input images themselves and are not consistent across varying conditions.

X. Discussion

The current method effectively calculates the rotationally-invariant displacement of a binarised crack image. The closer the binarised crack shape is to the original, the higher the accuracy of the results. Thus, the preprocessing parameters directly affect the performance of the algorithm. The current

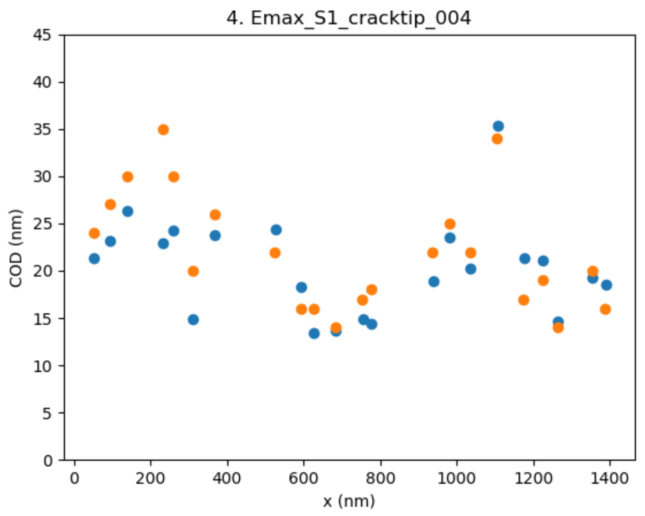


Fig. 8. Results of the algorithm with good thresholding

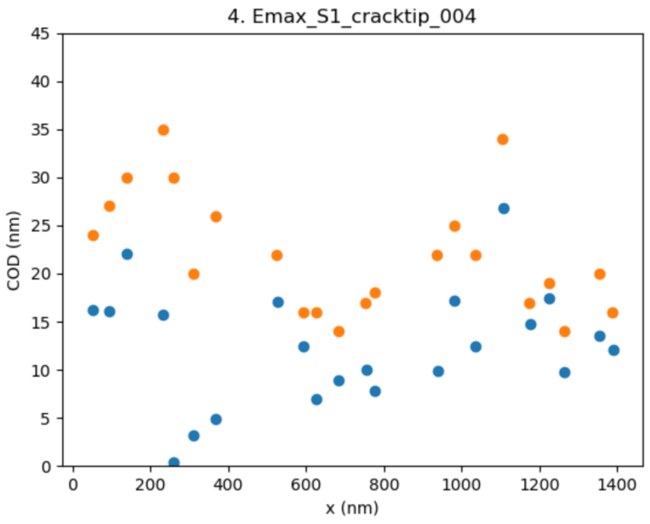


Fig. 9. Results of the algorithm bad thresholding

implementation still needs work in effectively determining the binarised crack shape. Some cracks in the dataset are defined by a black outline, but others are defined by white, so thresholding based on intensity is not adaptive enough for this dataset. In addition, shadows and noise such as those present in Fig. 10 affect the binarisation process, leading to poorly defined cracks and false positives outside of the main crack body. The binarisation algorithm will need a more generalised shape detection method to identify the crack edges before thresholding. However, while the preprocessing step requires human verification and input to find an effective outline, this may be automated once an adaptive outlining algorithm is implemented.

In addition to determining the COD, it may be possible to identify other elements of the crack using computer vision

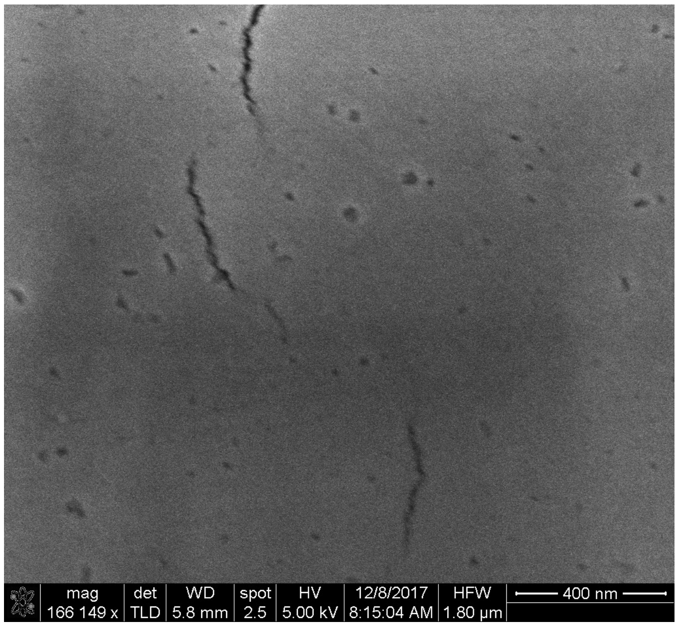


Fig. 10. Example of image with noise and unclear crack edges

techniques, such as the crack tip and length of the crack. However, to identify the crack tip, a considerably more sophisticated algorithm would need to be developed. The crack tip is identifiable mostly through context, rather than any features of its own. Within one image there may be many cracks of equal or larger size than the real tip, so it is necessary to follow the crack along its trajectory to find its tip. Context information may also be useful for removing outliers and identifying a more accurate COD.

Collecting data points through human input is time expensive. It also results in a limited selection of points that are prone to human error, are not reproducible, and require a higher error tolerance in evaluation. This method can automatically collect thousands of data points in a few seconds. With the increased amount of data, linear regression can be used to evaluate the change in position or displacement with more accuracy than before. With further improvements to the crack identification algorithms, this method of information extraction could provide results otherwise impossible to measure by hand, in an efficient, reproducible manner.

XI. Future Work

To meet the original project goals, further work remains to be done. This work will be focused on:

- Improving the preprocessing techniques and finding a robust method that will handle different conditions
- Creating new datasets which are suitable for generating a single linked dataset of a crack to allow querying along its length
- Further testing and evaluation
- Completion of the GUI and scripting interfaces

XII. Conclusion

While the project did not lead to a complete application, the goals defined in the project scope were met. The COD calculation algorithm was proven to work, with preprocessing being the determining factor in the results success. Available data was investigated, and a suitable format was found. Overall, the workflow was shown to be effective, and with further work, it is believed the application will be successful.

XIII. Acknowledgment

Data, including images and measurements, were provided by Carina Tanaka from the School of Mechanical and Manufacturing Engineering at The University of New South Wales. We are grateful for her generous assistance in explaining the data for us. Further, the idea for this project was developed with input from Professor Jamie J. Kruzic, also from the same department.

References

[1] H. Yi, C. Jingjie, L. Gang, *et al.*, "A new method of crack-tip opening displacement determined based on maximum crack opening displacement," *Engineering Fracture Mechanics*, vol. 78, no. 7, pp. 1441–1451, 2011.

[2] C. Shih, "Relationships between the j-integral and the crack opening displacement for stationary and extending cracks," *Journal of the Mechanics and Physics of Solids*, vol. 29, no. 4, pp. 305–326, 1981.

[3] S. Fünfschilling, T. Fett, R. Oberacker, *et al.*, "Crack-tip toughness from vickers crack-tip opening displacements for materials with strongly rising r-curves," *Journal of the American Ceramic Society*, vol. 94, no. 6, pp. 1884–1892, 2011.

[4] A. M. A. Talab, Z. Huang, F. Xi, *et al.*, "Detection crack in image using otsu method and multiple filtering in image processing techniques," *Optik-International Journal for Light and Electron Optics*, vol. 127, no. 3, pp. 1030–1033, 2016.

[5] T. Yamaguchi, S. Nakamura, and S. Hashimoto, "An efficient crack detection method using percolation-based image processing," in *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*, IEEE, 2008, pp. 1875–1880.

[6] M. Vidal, M. Ostra, N. Imaz, *et al.*, "Analysis of sem digital images to quantify crack network pattern area in chromium electrodeposits," *Surface and Coatings Technology*, vol. 285, pp. 289–297, 2016.

[7] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep learning-based crack damage detection using convolutional neural networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, 2017.

[8] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image segmentation using k-means clustering algorithm and subtractive clustering algorithm," *Procedia Computer Science*, vol. 54, no. 2015, pp. 764–771, 2015.