Challenges with feature detection in 3D point clouds. A Machine Vision Project

Simon Hansen-Møller

Department of Materials and Production, Aalborg University
Fibigerstraede 16, DK-9220 Aalborg East, Denmark
Email: sash13@student.aau.dk,
web page: http://www.mechman.mp.aau.dk/

Abstract
This project has been investigating the challenges of manufacturing feature detection using point clouds and image processing. Among used methods were a trigonometric approach and dot product in order to detect edge features. A flatness test was made to detect interior feature points. The obtained results proved that noise is a significant challenge which has to be overcome for any algorithm to perform well. Furthermore, while interior feature points were detected by use of a flatness test, it was not possible to determine the dimensions of the feature points. For this reason, it can be concluded that noise is a significant issue and that noisy points can be present within the point cloud itself while also be present as outlying points. While the outlying points can be removed using MatLab commands, these commands will not remove interior noise.

Keywords: Point clouds, Image processing, machine vision, Industry 4.0

1 General Introduction

Today's industry needs to be efficient and produce high quality products in order to keep up with competitors and stay in business. This means that modern factories need to convert their production to match industry 4.0 [1].

Among the needed technology for making an industry 4.0 conversion is a high degree of automation, efficient use of resources, and an intelligent production [2].

The intelligent production can be obtained by introducing machine learning and machine vision. These technologies are useful for the industry in numerous ways since they first of all allow for production systems to become more intelligent, but also allow for more processes to be automated. Through machine vision, machine learning can be implemented to allow robotics to perceive the surroundings which allows for industry 4.0 to be realised. This combination along with the integration high-speed networking, like 5G, then allows for new methods of low-waste production to be developed [3].

This paper will focus on the challenges with manufacturing feature detection in 3D point clouds and images, while also attempting to develop methods capable of detecting manufacturing features and their dimensions.

2 Fundamentals

2.1 Machine vision applications

As the idea of industry 4.0 is to create a smart manufacturing plant through the collection of data from different sensors. Machine vision contains numerous applications in the industry which can realise this idea of a smart manufacturing plant and thus, has the ability to improve the manufacturing process. Manufacturing companies that implement machine vision and control to their manufacturing processes often obtain an increase in productivity, reliability, and output [3].

2.1.1 Quality inspection

Quality inspection is a process in which the quality of an object or process is inspected. This could be the inspection of a weld made in an object. While traditionally performed by image processing, this task can also be performed using laser scanning. In this case, it is possible to detect the quality of welds down to a size under 2mm [4].

2.1.2 Quality assurance

Machine vision has become an integrated part of manufacturing with the introduction of industry 4.0. Traditionally, humans have had to make assessments of the quality of the products. This is however no longer necessary with the introduction of machine vision, as this technology allows for computers to collect data using 3D scanners and make decisions. This can be used in quality assurance in which machine vision can ensure precision, product quality, and reduction of operation losses from waste materials [3].

2.1.3 Conformity control

Conformity control is a type of quality control application which focuses on an object's correspondence with a theoretical model. This means that machine vision is being used to determine if an inspected object corresponds with its CAD drawing. Conformity control uses 3D point clouds to obtain information about the inspected area. In the case of comparing a theoretical model to a real object, the objective then becomes to determine if all of the 3D points are withing the expected tolerances of the theoretical model. This however results in a significant loss of accuracy and it is thus not possible to achieve a better accuracy than 0.5mm. The limited accuracy is due to the fact that the obtained measure is general rather than local [5].

3 Feature Detection

3.1 Robust feature detection

When capturing images in manufacturing, the photographed object may not always be photographed from the same angle since it may be rotated. For this reason, it is important that vision algorithms are robust as these algorithms are not sensitive to rotated objects or noise in the captured image [6].

3.2 Image feature detection

Image feature detection uses a camera for capturing a 2D image of an object. After this, the image is being processed by a computer in order to find features on the object. Several algorithms exist for detecting these features like Hough Circle, Shi-Tomasi and Canny Edge.

3.2.1 Hough Circle

The Hough Circle detector is a feature detection method capable of detecting the radius of a circle along with the centre of the circle while returning the information along with the coordinates for the circle centre. The detector works in two steps. The first step involves using a twostep Hough transform that is used to find the circle centre. The second step is a simple histogram step that identifies the radius of the circle. In order to locate the circle centre, it is determined that all vectors normal to the boundary of the circle, all must intersect at the circle centre. Using the Sobel operator, this can be estimated. Then, using the fact that a circle is described by the equation (1):

$$(x - a)^2 + (y - b)^2 = r^2 \tag{1}$$

A histogram processing is then computed as (2):

$$\delta = (x - a)^2 + (y - b)^2 \tag{2}$$

The largest produced peak using this equation is then determined as the circle radius [7].

3.2.2 Shi-Tomasi

The Shi-Tomasi is a method for corner detection he method mostly resembles that of Harrison corner however, the difference between the method lies in the scoring function where Shi-Tomasi will compute and use the minimum eigenvalues, as opposed to Harrison Corner [8].

3.2.2 Canny Edge

The Canny edge detector was developed in 1986. The edge detection is built upon three criteria.

1. Good detection.
2. Good localisation.
3. Only one response to a single edge.

The good detection criterion dictates that there should be a low probability for the edge detector to fail to detect real edges, similarly there should be a low probability for the detector to detect false edges. This means that the image in which the edge detection is performed should have a low amount of noise.

The good localisation criterion dictates that the points marked as edges should be as close to the centre of the true edge as possible.

Lastly, while the only one response to a single edge criterion is a part of the first criterion, multiple responses is not captured in it and thus a separate criterion for this is necessary [9].

3.3 Point cloud feature detection

3.3.1 Gauss map clustering

The Gauss map clustering method has been proposed by Weber et. al. [10] and was developed in 2010. The method uses a simple flatness test and Gauss Map Clustering. By computing the surface normal and determine if they are parallel, it can be determined that the surface is flat. After the flatness test, the remaining feature candidate points will undergo a second step in which the Gauss map is computed. In the case of a flat surface, the Gauss map will represent one cluster. However, in the case of a curve or a tangent plane discontinuity, the Gauss map will be behaving differently [10].

3.3.2 Centroid method

The centroid method is built upon the use of k-nearest neighbour search method and it can be assumed that this point is located in the centre of the k-nearest neighbours. Then, by computing a centre point, called the centroid, it can be determined if this is in fact the case. As the point nears the edge, the nearest points will start drifting in the direction along and away from the edge, thus the centroid will start drifting as well. Then, when the centroid as drifted as certain length away from the search point, it can then be determined that the search point is in fact an edge point. The corners are defined as the intersection between two or more lines, thus using the data obtained about the edges, the corners can be found. To begin with, the authors compute the surface normals of the point cloud and the curvature of the edge points. The curvature of the edge points is computed as the eigenvector e of the smallest eigenvalue λ_1. This is determined from the covariance tensor Σ_E (3) of the neighbouring points.

$$\Sigma_E = \frac{1}{\Sigma(R-d_i)} * \sum_{i=1}^{n}(R-d_i) * (p_i - p_c) * (p_i - p_c)^T \forall \mathrm{p} \in \mathrm{E} \qquad (3)$$

Here, R is a predefined radius, di = ||pi − p||2 and pc is the centroid of the neighbouring points. The reason as for why the authors uses this tensor lies in the advantage that it assigns a smaller weight to the distant points such that it increases the repeatability in the presence of clutter. Lastly, a check for the type of corner is necessary. This check determines if the corner is the intersection of two or more corners as this is a possibility in 3D space. Thus the check is performed by evaluating the maximum variation in euclidean distance within x, y, z space [11].

3.4 Noise

Noise is a challenge faced by both image processing and laser scanning. Image noise typically emerges from the object of which the image has been captured. The noise can emerge due to the object material which reflects light and thus disturbs the camera, or it can emerge from the image background which may possess a similar issue. The colour of an object may also present challenges for cameras. As an example, it may be challenging for an image processing algorithm to detect the full size of an object if it has a dark colour. The reason for this lies in the fact that the lighting of the image could cast shadows from the object which challenge the algorithm as it cannot detect the difference between the shadow and the object itself.

As mentioned, noise is also a problem for laser scanners. Here noise is found in form of missing points or points that are invalid, see Figure 1. The invalid points may be points that are "not-a-number" or points that are not a part of the object itself.

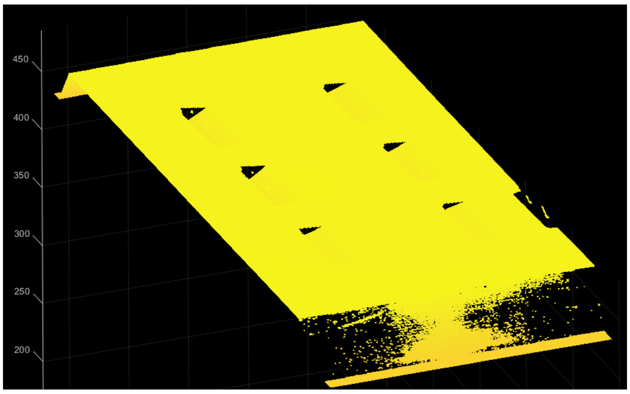


Fig.1. Example of noise in a Point cloud.

When it comes to the task of feature detection, it is evident that noise, no matter its form, will present a challenge as it may disturb the task and make it difficult and in some cases impossible to carry out the task. Therefore, noise is a challenge that must be taken into consideration and dealt with in order for feature detection to be possible.

3.4.1 Noise removal

Noise removal is an import part of the pre-processing step of any image or point cloud processing algorithm. Without noise removal, the algorithm will surely fail its purpose. Numerous methods for noise removal exists, thus two often used methods will be described.

<u>Gaussian blur:</u>

Gaussian blur is a function often used to reduce noise in image processing. The function uses a kernel described by a gaussian function to normalise and remove noise in an image.

<u>Point cloud:</u>

For noise removal in point clouds, MatLab has a command for removing spare outlying points. This is done by computing the mean μ and standard deviation σ of the nearest neighbour distances.

4 Solution

4.1 Point cloud feature detection

For the Point cloud solution, a laser scan was provided to the author, see Figure 1. This laser scan features a point cloud of nearly 2.3 million points. A point cloud this size means that some computations will be rather heavy for a computer to compute which creates a disadvantage.

While it was not possible for the author to develop a method for detecting features in 3D point clouds, focus will instead be put on the methods used to attempt to find a solution. These attempts cover: Trigonometric, dot product, and line analysis.

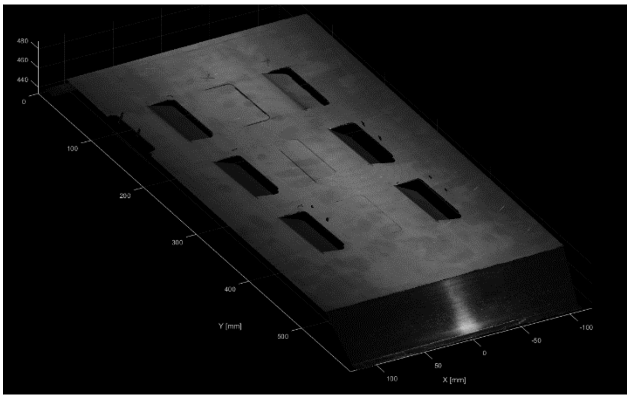


Fig.2. Point cloud object.

All solutions featured a flatness test that was modified and therefore performed differently between the solutions. Common for the flatness tests were the computation of surface normal. What differed was in

4.1.1 Trigonometric method

The trigonometric attempt uses the in-build functions in MatLab known as boundary and knn search (k-nearest neighbour). The boundary function finds the boundary of a point cloud and returns these points in an array which can then be used later. This boundary is a thin line that contains one layer of points. Using this, it was assumed that the two nearest neighbours to each point would be located on the left and right side of the point. Under this assumption, it was determined that by using simple trigonometric mathematics, the angle between the obtained three points could be obtained. The reason for this is found in the fact that the angle between the points will always be possible to calculate, and in the case of using a knn search in MatLab, it is possible to obtain the indices and the distance between the nearest points. It is then a simple matter of computing the distance between the two nearest points themselves as MatLab provides the distance between the search point to its nearest points.

By using this method, corner points should be possible to find as they would have an angle of approximately 90 degrees, and bend points would have an angle above 90 degrees or below.

Lastly, the interior feature points would also have to be found. By drawing inspiration from the Gaussian Clustering method, a flatness test was performed. Here, the algorithm started by first finding the interior points of interest. These points were found by computing the surface normals. Then, by using the same scheme as before, using the nearest neighbour search, the six nearest surface normals was found for each surface normal. By computing the angle between these normals, interior points of interest could be found. These points of interest were defined as the points in which the angle between the surface normals exceed 1 degree as this would indicate a change in the object structure.

4.1.2 Line analysis

The line analysis method attempted to analyse the data set two points at a time. The method assumes that the points in the point cloud are placed in a straight line and thus the point are moving along the x or y axis in lines. This assumption dictates that it would be possible to detect edges and corners of the scan, seeing as it can be assumed that the first scanned point is a corner, and the same can be assumed for the last point. The last corners could then be detected using the dot product. This would once again require the use of MatLabs boundary command in which difference vectors between the nearest points can be calculated. Using the fact that in case the dot product is 0, the vectors will be perpendicular, the corners can be detected. The reason for this is found in the fact that in case the vectors are perpendicular, it would be a reasonable assumption that two lines are meeting in the search point.

4.1.3 Dot product method

The dot product method attempted to detect edge features by using the dot product. The idea behind the method is centred around the fact that two vectors are perpendicular in the case that the dot product is 0. By using this knowledge, it is possible to compute difference vectors from a search point to its nearest neighbour, not unlike the method used in the trigonometric approach.

By using the same flatness test as before, the dot product method attempts to make the method more accurate. The trigonometric approach used the nearest normal vectors, as it was assumed these would be the same as the nearest points, whereas the dot product method uses the nearest points and then find their normal vectors, using the index number. The reason for this is due to the assumption that the indexes for the normal vectors and the points are the same.

4.2 Image feature detection

For the image processing solution, two sheet metal objects were created. These objects were slightly different in their presence as the first features two circular holes of respectively 10 and 8 mm diameters, whereas the other featured only one of 10 diameter. Both of the objects had a square hole with the sides measuring 10 mm, giving it an area of 100 mm^2. Similarly, both objects had a large bend, with the other having an extra bend. As these objects were manufactured by hand, the angles in the bends were unfortunately not measured.

The image solution used several methods in order to obtain its results. These methods included Houghes circles, Shi-Tomasi, and BLOB analysis.

BLOB analysis was used in order to find the square. Here the image was converted to a binary image, meaning it was converted to black and white. The generated shapes were then analysed after which the square could be found.

4.2.1 Developed method

The developed computer vision program works by first loading the two input images which are converted to grey scale images. Processing the first image, the program starts by finding the circular hole. It does this by using the Hough circle method. Here Hough gradient method is used. The dp value is put to 1, and the minimum distance to the centres of the circles is put to 20, since this value is unimportant due to the fact that there is only one circle in the object. the two parameter values are put to 50 and 30 respectively as this should make the algorithm work as intended. As the circular hole has a diameter of 10mm, the maximum and minimum radius are put to 40 and 12. The reason for this choice is to find all possible holes. It should also be kept in mind that these values are not in mm, but in pixels. Following the finding of the circular hole, the program moves on to find the square hole. Here, the maximum and minimum areas will have to be defined in order to detect all possible square holes. After this, edge detection using canny edge is used to find all of the edges. Using this method, contours are drawn, and areas of the contours are calculated.

Using the minimum and maximum areas, the square is found as the area between these values. By finding the circular and square hole features, the algorithm will now have to find the corners using Shi-Tomasi and the angle between the object and its bend part. By calculating the Euclidean distance between the points, it is possible to determine which are points of interest.

To determine if the hole features are located in their correct places, the distance from the features to the corners is also computed.

5 Experiments

The conducted experiments took root in 4 requirements:

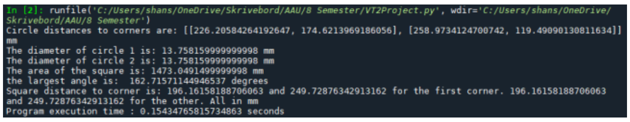
1. Tolerances: The algorithm should mark products that exceeds the features of the CAD drawings by 0.5 mm. This means that holes, deviating from the CAD drawings by 0.5 mm should be marked as defect.
2. Features: The algorithm must be able to detect features both interior and on the edge of an object.
3. Materials: The algorithm should be able to perform feature detection on various materials such as steel and aluminium.
4. Robust: The algorithm should be robust and should be able to detect features on multiple objects and should not be sensitive to the processed object being rotated on the image.

As the developed point cloud methods were unable to compute the dimensions of any features, they will not be a part of the experiments. Instead, only the image processing solutions will be a part of them.

5.1 Experiment 1, Tolerances

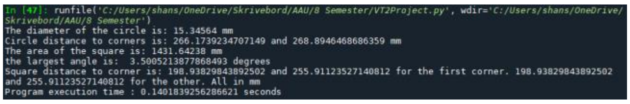
The image processing code was able to find the circular hole features along with the square hole and some of the corners.

Unfortunately, the images were of a poor quality due to the background which interfered greatly with the corner detection method. This was anticipated, though the extend of the interference was not.



```
In [2]: runfile('C:/Users/shans/OneDrive/Skrivebord/AAU/8 Semester/VT2Project.py', wdir='C:/Users/shans/OneDrive/Skrivebord/AAU/8 Semester')
Circle distances to corners are: [[226.20584264192647, 174.6213969186056], [258.9734124700742, 119.49090130811634]] mm
The diameter of circle 1 is: 13.758159999999998 mm
The diameter of circle 2 is: 13.758159999999998 mm
The area of the square is: 1473.0491499999998 mm
the largest angle is:  162.71571144946537 degrees
Square distance to corner is: 196.16158188706063 and 249.72876342913162 for the first corner. 196.16158188706063 and 249.72876342913162 for the other. All in mm
Program execution time : 0.15434765815734863 seconds
```

Fig.3. Results obtained for the first object.



```
In [47]: runfile('C:/Users/shans/OneDrive/Skrivebord/AAU/8 Semester/VT2Project.py', wdir='C:/Users/shans/OneDrive/Skrivebord/AAU/8 Semester')
The diameter of the circle is: 15.34564 mm
Circle distance to corners is: 266.1739234707149 and 268.8946468686359 mm
The area of the square is: 1431.64238 mm
the largest angle is:  3.5005213877868493 degrees
Square distance to corner is: 198.93829843892502 and 255.91123527140812 for the first corner. 198.93829843892502 and 255.91123527140812 for the other. All in mm
Program execution time : 0.1401839256286621 seconds
```

Fig.4. Results obtained for the second object.

As seen in Figures 3 and 4, the detected dimensions are not those of the actual manufactured product, thus this requirement is vied as Failed.

5.2 Features

As stated, the algorithm correctly detects all features, thus the requirement is viewed as Passed.

5.3 Materials

Though as it was intended, objects of different materials were not manufactured as time ran short. Thus, this requirement was not fulfilled and is viewed as Failed.

5.4 Robust.

The algorithm was able to detect all features, even given awkward angles and poor image quality, thus this requirement is viewed as Passed.

6 Discussion

Feature detection is an important task of any manufacturing vision system as it allows for the analysis of features and determine if these are within given product development tolerances. The process can be carried out using either image processing libraries like OpenCV, or it can be done using laser scanning and point clouds. Point clouds are highly complex as noise is a challenge that has to be overcome as it can put an end to even the most analytical methods. This means that a robust algorithm for feature detection in point clouds has to take all of these things into account in order to yield the desired results, which is no small feat. The noise in point clouds can be found as either the lack of points or extra points, as discovered during the development of point cloud methods, see section 4.1. As it was shown, these extra points, or lack of points, interfere with the algorithms so that a flatness test cannot be performed properly. Thus, the issue of noise is a significant one that has to be solved for any point cloud processing algorithm.

6.1 Future work

In order to develop a proper point cloud processing method, this project will need to overcome all of the detected challenges. This means finding a method to counteract and remove noisy points, finding a method for detecting corner and bend points and finally, compute information about the feature points.

The author of this paper proposes three methods which may achieve the desired outcome for feature detection:

1. Computation of points within a given distance
2. Conversion of points to pixels
3. Modified centroid method

6.1.1 Computation of points within a given distance

Compute number of points within a certain distance for all points. By making this analysis, it will be possible to determine if a point is on an edge, corner, or bend. This method is not unlike that of the centroid method, and depending on the number of points, may also help determine what points are noise (if step size is 0.5, if a point is then within 0.06, the point is likely noise). It will then be possible to define a 3D matrix grid and use this grid to analyse the point cloud for feature points.

6.1.2 Conversion of points to pixels

By converting the points in the point cloud to pixels, it may be possible to determine the feature points using ordinary computer vision methods. In order to do this, it will be necessary to convert the z-axis of the point cloud to grey scale, thus converting the problem from a3D problem to a 2D problem. The reason is that the gray scale value can be viewed as a intensity value and the point cloud is now only in x and y dimension. In areas with no points, the gray value would be 0.

Combined with the method of computing points within a given distance, it will be possible to detect edges and corners, by using gradient or Laplacian methods. Here the gradient method is the first derivative of the pixel, while the Laplacian is the double derivative. In principle, both methods can be applied to detect edges, however they have their advantages and disadvantages. The Laplacian method may yield a better result, however it will be more sensitive to noise. The gradient based method may not yield as good a result as the Laplacian, however it will not be as sensitive to noise either [12].

6.1.3 Modified centroid method

The centroid method holds a potential that can be used in the fact that it computes a centroid vector. While the authors behind the centroid method used it to detect edges and define lines which then detected corners, it is possible to modify the centroid method and combine it with the dot product method to find corners. In case the centroid vectors are parallel, the search point is classified as an edge. However, in case it is not, the point can be classified as a corner or a bend point. Then using this, it will be possible to compute planes and find the angle between the planes and thus, find the dimensions of possible features. This can be used for all points in the point cloud, though caution should be used in noisy areas.

7 Conclusion

Industry 4.0 needs efficient and accurate methods for processing acquired data in order to keep the manufacturing process efficient and produce high quality goods. Feature detection is an important part in the process of quality assurance and control like conformity control. To use feature detection, it is necessary to use machine vision with efficient computer vision software. By using point clouds, high accuracy feature detection can be obtained, though noise is a significant obstacle that has to be overcome in order for any algorithm to be successful.

This project attempted to develop several methods for feature detection in point clouds but, due to noise, was unable to develop any methods that allowed for features and their dimensions to be detected and described. Using Python and OpenCV, it was possible to develop a method for feature detection. The developed algorithm was capable of detecting features and describe their dimensions, though the algorithm did not perform optimally due to poor image quality, stressing the issue of using images for feature detection.

Therefor it can be concluded that feature detection is no easy task when done using point clouds, however by having poor image quality, robust algorithms that produces accurate results is not an easy task either.

Acknowledgement

The authors of this work gratefully acknowledge Grundfos for sponsoring the 8th MechMan symposium.

References

[1] Aysegul Yilmaz et al. Lean and industry 4.0: Mapping determinants and barriers from a social, environmental, and operational perspective. Nov. 2021. doi: 10.1016/j.techfore.2021.121320.

[2] Hyoungkwan Kim. Chapter 6 - Use Cases for Contractors. url: https://www.sciencedirect.com/topics/engineering/defect-detection. (accessed: 18-02-2022).

[3] Mohd Javaid et al. Exploring impact and features of Machine Vision for progressive Industry 4.0 culture. Nov. 2021. doi: 10.1016/j.sintl.2021.100132.

[4] Lee BR. Nguyen HC. “Laser-vision-based quality inspection system for small-bead laser welding.” In: International Journal of Precision Engineering and Manufacturing 15 (2014), 415–423. doi: https://doi.org/10.1007/s12541-014-0352-7.

[5] Victor Alonso et al. “Industry 4.0 implications in machine vision metrology: an overview”. In: Procedia Manufacturing 41 (Jan. 2019), pp. 359–366. doi: 10.1016/j.promfg.2019.09.020.

[6] Michael S. Lew Nicu Sebe. Robust Computer Vision. url: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.4.1783&rep=rep1&type=pdf. (accessed: 08-02-2022).

[7] J. Illingworth and J. Kittler. “The Adaptive Hough Transform”. In: IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9.5 (1987), pp. 690–698. doi: 10.1109/TPAMI.1987.4767964

[8] OpenCV. Shi-Tomasi Corner Detector Good Features to Track. url: https://docs.opencv.org/4.x/d4/d8c/tutorial_py_shi_tomasi.html. (accessed: 12-04-2022).

[9] John Canny. “A Computational Approach to Edge Detection”. In: IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-8.6 (1986), pp. 679–698. doi: 10.1109/TPAMI.1986.4767851.

[10] Christopher Weber, Stefanie Hahmann, and Hans Hagen. Sharp feature detection in point clouds. 2010. doi: 10.1109/SMI.2010.32.

[11] Syeda Mariam Ahmed et al. Edge and Corner Detection for Unorganized 3D Point Clouds with Application to Robotic Welding. 2018. doi: 10.1109/IROS.2018.8593910.

[12] Xin Wang. “Laplacian Operator-Based Edge Detectors”. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 29.5 (2007), pp. 886–890. doi: 10.1109/TPAMI.2007.1027.