UNIVERSITY OF BIRMINGHAM
University of Birmingham Research at Birmingham

Automatic identification of mechanical parts for robotic disassembly using the PointNet deep neural network

Zheng, Senjing; Lan, Feiying; Baronti, Luca; Pham, Duc; Castellani, Marco

DOI:
10.1504/IJMR.2022.10039082

License:
None: All rights reserved

Document Version
Peer reviewed version

Citation for published version (Harvard):
Zheng, S, Lan, F, Baronti, L, Pham, D & Castellani, M 2022, 'Automatic identification of mechanical parts for robotic disassembly using the PointNet deep neural network', *International Journal of Manufacturing Research*, vol. 17, no. 1. https://doi.org/10.1504/IJMR.2022.10039082

Link to publication on Research at Birmingham portal

General rights
Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

•Users may freely distribute the URL that is used to identify this publication.
•Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
•User may use extracts from the document in line with the concept of 'fair dealing' under the Copyright, Designs and Patents Act 1988 (?)
•Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy
While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.

Download date: 28. Apr. 2024

Automatic Identification of Mechanical Parts for Robotic Disassembly Using the PointNet Deep Neural Network

Senjing Zheng[1], Feiying Lan[1], Luca Baronti[2], Duc Truong Pham[1], Marco Castellani[1*]

[1] Department of Mechanical Engineering, University of Birmingham
[2] School of Computer Science, University of Birmingham
{sxz595, fxl655}@student.bham.ac.uk
{d.t.pham, m.castellani, l.baronti}@bham.ac.uk

***Abstract*—Identification is the first step towards the manipulation of mechanical parts for robotic disassembly and remanufacturing. This paper presents a case study on the identification of objects from 3D scenes (point clouds) of mechanical components of automotive devices. The identification task is carried out through PointNet, a recently developed deep neural network system. PointNet is capable of identifying objects irrespective of their position and orientation in the point cloud. In this work, PointNet was used to recognise twelve instances of parts of different turbocharger models for automotive engines. The training instances consisted of different types of mechanical parts, as well as different models of the same type of part. Point clouds of partial views of the parts were created from CAD models using a purpose-developed depth-camera simulator. Different levels of sensor imprecision/noise were simulated. The results of the tests indicated that PointNet can be trained to recognise with good accuracy the various mechanical objects, and that its learning procedure is consistent and effective. In presence of sensor imprecision, the recognition accuracy in the recall phase can be increased adding some stochastic error to the training examples. The possibility of training twelve independent classifiers to be employed separately or in one ensemble classifier was also investigated. The accuracy results were comparable to those obtained using one classifier for all the parts.**

Keywords: Remanufacturing; Disassembly; Automotive; Machine Vision; Point Clouds; Deep Neural Networks.

Acknowledgments: *This work was funded by the UK Engineering and Physical Sciences Research Council (EPSRC), Grant No.EP/N018524/1 - Autonomous Remanufacturing (AutoReman) project.*

I. Introduction

Remanufacturing is an environmentally sustainable product end-of-life (EOF) strategy (Kin et al., 2014) different from recycling, repair and refurbishing. It restores products at their end of service life to at least original performance via a combination of re-processing and substitution of used components (Johnson & McCarthy, 2014). In doing so, remanufacturing makes EOF products available for another complete life cycle (Ijomah, 2002).

The first step in remanufacturing is to disassemble the returned products into the individual parts for processing. Disassembly is not only relevant to remanufacturing, but also service operations (maintenance and repair). Disassembly is to date the bottleneck in remanufacturing processes, due to its labour intensive and time-consuming nature, the possibility of safety hazards to the operators, and the lack of effective automatic methods (Liu et al., 2018). Differently from assembly, where the features, location, and pose of the parts are known, the task of disassembling returned products is fraught with uncertainties on the integrity and state of the parts. The nature of such uncertainties may vary from minor scratches to corrosion, physical damage, or complete removal of a component.

This paper focuses on the creation of a vision system for automated disassembly of mechanical parts for remanufacturing. Safe and reliable robotic manipulation of an object is possible only if it has been correctly recognised and located. This calls for reliable machine vision and object recognition systems.

Due to the variability in appearance and state of used parts, traditional 2D feature-based methods often give poor performances on object recognition tasks for disassembly applications. For example, 50% accuracy was reported by Wegener et al. (2015) on an application involving robot assisted disassembly of electric vehicle batteries, whilst Vongbunyong et al. (2013a) reported 64% accuracy in a similar robotic disassembly application. The main cause of inaccuracy in standard 2D machine vision methods is the loss of structural information about the objects. For this reason, 3D models (scenes) are currently investigated.

A 3D model of an object can be formed merging two 2D camera images from different perspectives. However, this method is limited by problems such as the field of view of the individual cameras, and the difficulty of matching multiple perspective images (Hussmann et al., 2008). An alternative is to use structured light to obtain 3D information directly by a projector and a receiver (Salvi et al., 2004). This method is highly precise but requires an expensive apparatus. A low-cost time-of-flight depth camera can be used to provide auxiliary high-resolution information to recover depth information in 2D camera images. Regardless of the acquisition method, 3D images are usually displayed using data structures called *point clouds*. Point clouds have been widely used in 3D reconstruction, recognition, and semantic segmentation (Rusu & Cousins, 2011).

Point clouds are typically composed of millions of data points, each point being represented through its X-Y-Z coordinates. Due to their large size, and the topological relationships

*corresponding author

between the points in the model, point clouds are difficult and lengthy to process and understand.

Amongst many machine learning methods developed for pattern recognition in point clouds, the deep neural network (DNN) PointNet (Qi, Su, et al., 2017) showed great promise for 3D scene classification and segmentation. PointNet takes point clouds directly as input, and is able to recognise objects and their components irrespective of position and orientation. This feature makes PointNet an ideal candidate for object recognition in a highly unstructured domain such as the disassembly and manipulation of returned parts for remanufacturing.

Like all neural networks, PointNet needs to be shown a large set of examples during the training phase. In disassembly applications this would entail scanning several instances of each mechanical part. Each point cloud would need to be formed out of several partial captures from different perspectives. This would require a considerable effort which would need to be repeated every time a new part is introduced.

To address this problem, PointNet was trained on point clouds generated from CAD models of the mechanical parts. For many manufactured products, CAD models are usually available or obtainable. A time-of-flight depth camera simulator was developed to create realistic point clouds of a number mechanical parts from CAD representations. The simulator allows perturbing the original CAD models with noise, and the ability of PointNet to recognise the mechanical parts was tested under different noise levels. The tests aimed to provide a proof of concept of the possibility of training PointNet to recognise complex mechanical parts, using point cloud scenes generated from CAD representations, and employing PointNet in noisy real-life conditions.

The key contributions of this study are:

- The recently developed PointNet deep neural network was for the first time applied to the recognition of complex mechanical parts from point cloud models;
- PointNet was trained on point cloud scenes generated from CAD models using a purpose-developed depth-camera simulator, and tested on scenes containing different levels of sensor imprecision from those used in the training examples;
- The robustness of the training procedure was evaluated for different levels of simulated sensor noise. To the best of the author's knowledge, this is the first time the noise rejection capabilities of PointNet are evaluated;
- The study showed that perturbing the training patterns with a moderate amount of noise significantly improves the noise rejection capabilities of PointNet

The paper is structured as follows: Section II reviews the literature on machine vision and remanufacturing. Section III introduces the proposed method for automatic identification of mechanical parts. Section IV presents the experimental results, which are discussed in Section V. Section VI concludes the paper.

II. Background

The interest in remanufacturing and automated disassembly is shown by a rapidly growing literature, where effective machine vision algorithms are fundamental elements in flexible automated disassembly processes. The standard solutions employ object recognition methods based on 2D images. Vongbunyong et al. (2013a,b, 2015) used cognitive robotics to emulate human intelligence for automated disassembly of electronic components. In this application, a rule-based vision system is used for recognition and localisation of the components, and for obstacle detection in trajectory planning. Wegener et al. (2014, 2015) proposed a robotic assisted system to disassemble electric vehicle batteries, using machine vision for detection of screws. Similarly, Bdiwi et al. (2016) used machine vision for screw detection in an automatic disassembly workstation.

Due to the limitations in accuracy of 2D vision systems, some authors employed 3D systems for object identification. Umeda & Arai (1996) proposed a vision system prototype for mechanical assembly and disassembly processes. The system employed high-speed range image sensing for 3D perception, and primitive surface features such as planes and cylinders for model matching. A multi-source heterogeneous vision perception framework for human-robot collaboration was proposed by Yang et al. (2018). The system utilised RGB-D cameras to capture the 3D structure of the working area, and binocular cameras to track the worker's hands. A semi-automatic non-destructive disassembly disassembly cell was proposed also by Torres et al. (2004).

The standard implementation of machine vision in industrial applications is based on template matching of manually generated feature descriptors from 2D images. One of the major limitations of template matching is often the computational complexity of the feature extraction methods, which is problematic for real-time implementation. Lowe (1999) proposed the popular 2D scale and rotation invariant feature descriptor SIFT (Scale-invariant Feature Transform), providing a robust feature detection technique for 2D images. Image classification was based on nearest-neighbour indexing for the identification of candidate object matches, and a low-residual least-squares the verification of the candidate matches. Bay et al. (2006) proposed SURF (Speeded-up Robust Features), an improvement of SIFT descriptors which are faster to compute than their predecessors. Despite the improvements in speed, the computational complexity of SURF is still a challenge for real-time implementation in industrial applications. Rublee et al. (2011) proposed a real-time feature extraction technique called ORB, and performed object recognition performing brute-force matching with stored templates.

Feature descriptors were also extended to 3D models for object identification and matching in point clouds. For example, Rusu et al. (2008) proposed the persistent feature histogram method to extract a set of robust features for point cloud registration. Other types of descriptors include information on the structural neighbourhood of the individual points (Chua & Jarvis, 1997).

In recent times, machine learning approaches have been at the core of many machine vision solutions for manufacturing and re-manufacturing applications. Machine learning is particularly useful because it obviates the need of the time-consuming manual feature extraction and selection

steps performed in standard template-based pattern recognition techniques (Carlevaris-Bianco & Eustice, 2014; Zagoruyko & Komodakis, 2015; Xing et al., 2018; Dai, 2019; Schonberger et al., 2017). Machine learning can be used for proper feature selection, namely to select a fully descriptive minimal subset of elements from a set of pre-defined candidates (Castellani & Rowlands, 2008). Alternatively, machine learning can be used to train an image pre-processing block to automatically extract features. A learning classifier is then cascaded to the pre-processing block. This latter arrangement constitutes the basis of the widely successful Convolutional Neural Networks (CNNs).

A CNN is a form of DNN (LeCun et al., 2015), namely an artificial neural network characterised by more than two layers of processing units. The initial block of pre-processing units is usually composed of convolution layers alternated by *max-pool* layers. The convolution layers are filters trained to detect features in the input image or point cloud, and propagate this information to the next layers. The *max-pool* layers are used to perform a *downsampling* of the features. In some cases, the initial block is instead composed of layers of standard perceptron (Pham et al., 2007) units that are used to extract features from the input images or point clouds. In the last decade, DNNs and CNNs in particular found wide application in machine vision (Szegedy et al., 2016), specially for image classification tasks (Krizhevsky et al., 2012).

An example of DNN application to the manufacturing field is the DNN system developed by Krüger et al. (2019). The DNN is used for identification of mechanical parts, where multiple pictures of a single mechanical component are taken from different angles. The pictures are then analysed and classified with a CNN. This approach showed promising results, even though using a series of 2D pictures to approximate a 3D representation may lead to missing useful object features. The process of image capture took half a year work. A CNN was also used by Weimer et al. (2016) for optical defects detection in the field of industrial inspection.

An instance of DNN that was specifically designed to process point cloud models is PointNet (Qi, Su, et al., 2017). PointNet can be used for object recognition as well as for object segmentation. In the latter case, the segmentation task is approached as pointwise classification. PointNet is composed of several stacked layers of processing units, including three full MLPs: two used for point cloud pre-processing, and one for the final classification step.

An important feature of PointNet is that the identification result is invariant to rigid transformations of the input point cloud. That is, PointNet can recognise objects irrespective of their position and orientation in the 3D image, removing the need time-consuming model segmentation and object re-orientation. The PointNet architecture will be described in more detail in section III-B.

PointNet has been successfully used in different fields such as semantic segmentation (Garcia-Garcia et al., 2017), including indoor and outdoor large-scale spatial contexts (Engelmann et al., 2017). Guerrero et al. (2018) employed PointNet to extrapolate local geometric properties like normals and curvatures from noisy point clouds. One of the primary uses of PointNet is 3D object detection. For this purpose, Zhou & Tuzel (2018) trained PointNet to identify cars, pedestrians, and cyclists in urban scenes in LiDAR point clouds. The point clouds were partitioned in a 3D grid of voxels that was fed to the DNN. Qi et al. (2018) used instead RGB-D data to train PointNet to recognise objects in indoor and outdoor scenes.

The latest development of PointNet is PointNet++ (Qi, Yi, et al., 2017), a system that recursively applies PointNet on a nested partitioning of the 3D scene. This technique is able to exploit the non-uniform density distribution of points to extract features in an hierarchical fashion, achieving state-of-the-art performances on different benchmarks.

III. Methodology

A. The Recognition System

The aim of this study is to build a robust recognition system that is able to learn from CAD models to identify mechanical objects for disassembly purposes. This system needs to be tolerant to sensor noise, and reasonable alterations of the objects due to wear, tear, and removal or substitution of their components.

The recognition system will constitute the first block of the overall disassembly system. Once an object has been recognised, its pose can be recovered (Besl & McKay, 1992; Quan et al., 2018) and robotic manipulation can be planned. Furthermore, the mechanical object can be segmented into its components for disassembly sequence planning.

The proposed model recognition system is based on the use of the PointNet DNN architecture. The main advantages of PointNet are the following: it was built to be directly used on point clouds without the need of time consuming model pre-processing; it is able to recognise patterns irrespective of their location and orientation; and like all ANN systems is expected to possess intrinsic generalisation capabilities that make it robust to sensor noise and fair variations of the object. In addition, PointNet is also suitable for object segmentation, which will be implemented at a later stage.

The first step in any machine learning application is the creation of a reasonably large data set of examples for the training and validation of the learning system. Although there are online available collections of 3D point cloud models (Deng et al., 2009; Lin et al., 2014; Krasin et al., 2017), none of them contains mechanical objects.

In this study, CAD models of mechanical parts were used to generate the sets of point cloud scenes used for training and testing the PointNet. CAD models of mechanical parts are often easily obtainable from the manufacturer, or can be created via reverse engineering. The point clouds were generated from the CAD models via a purpose-developed software that simulates the 3D image capture process in a real scene, and creates a 3D model (point cloud) of the object.

The goal is to prove that PointNet can be trained to recognise with good accuracy complex mechanical parts from point clouds, and is able to generalise the learned knowledge to scenes presenting the objects in previously unseen orientations and in presence of realistic levels of noise.

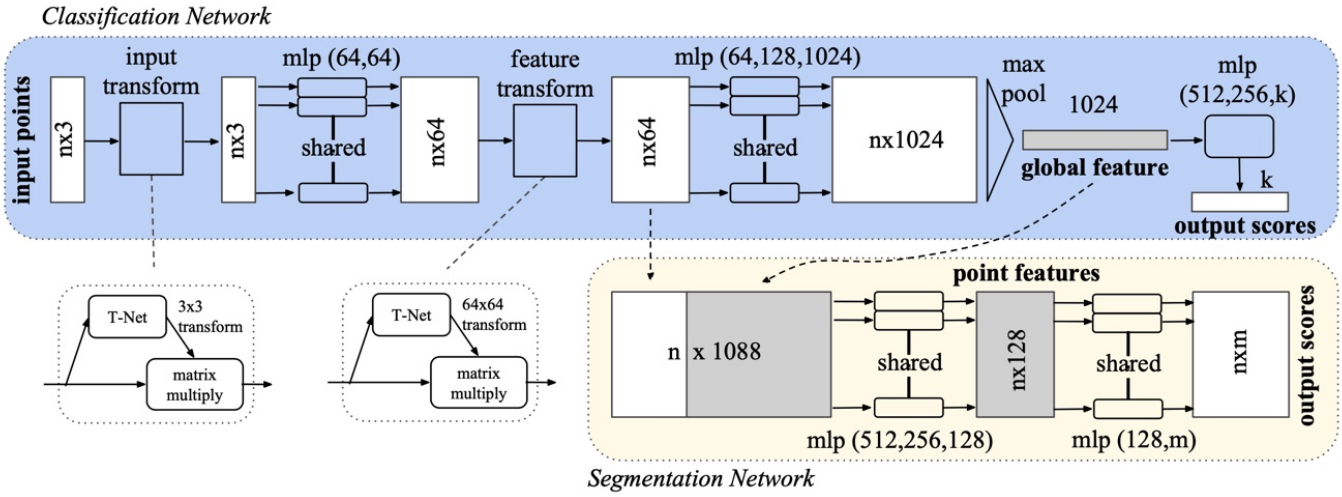


Fig. 1: PointNet Architecture by Qi, Su, et al. (2017)

B. The PointNet Deep Neural Network

PointNet is a DNN architecture specifically designed to perform classification and segmentation of point cloud models (Qi, Su, et al., 2017). It is structured as a pipeline of several neural layers as shown in Figure 1 , and is composed of three modules: the max pooling layer, a local and global information combination structure, and two joint alignment networks.

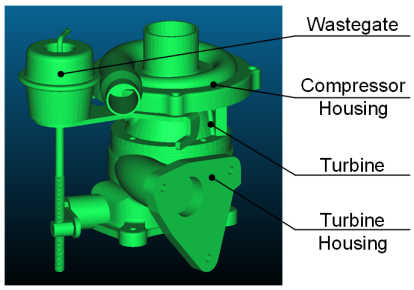
The first module (max pooling layer) is composed of a set of multi-layer perceptrons (MLPs) (Pham et al., 2007) and a max pooling function. It implements a symmetric function that aggregates the information from the individual points in the cloud, and is invariant to the order of the points and rigid transformations of the object. That is, the first module implements a series of rotation and translation invariant feature detectors.

The second module (local and global information combination structure) aggregates local and global information on the point cloud. This is useful for some tasks (e.g. scene segmentation) that require both the local and global features of the points. It concatenates the global information to the point features, and processes this information in order to generate the output.

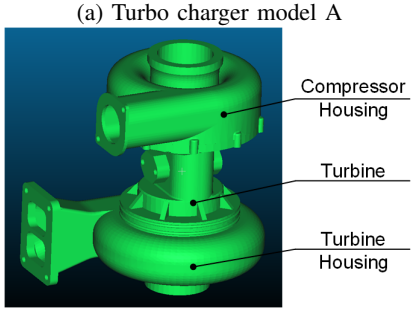
The third module (joint alignment networks) ensures that the semantic labelling of the point cloud is invariant to rigid transformations. Instead of converting all the points to a canonical space (Jaderberg & Simonyan, 2015), PointNet uses a mini-network (called T-Net) to predict the affine matrix describing the rigid transformation. This network is trained along all the other components, and the resulting transformation matrix is applied directly to the coordinates of each point. The same module is also duplicated to align features from different input point clouds.

C. The 3D Model Set

The point cloud used in the experiments were created from CAD models of the two types of turbochargers (henceforth called *model A* and *model B*) shown in Figure 2. The turbochargers had a very similar structure, containing a compressor housing, turbine, and turbine housing. Model A included an additional wastegate. In total, the two turbochargers could be disassembled into the twelve different types of components shown in Figure 3.



(a) Turbo charger model A

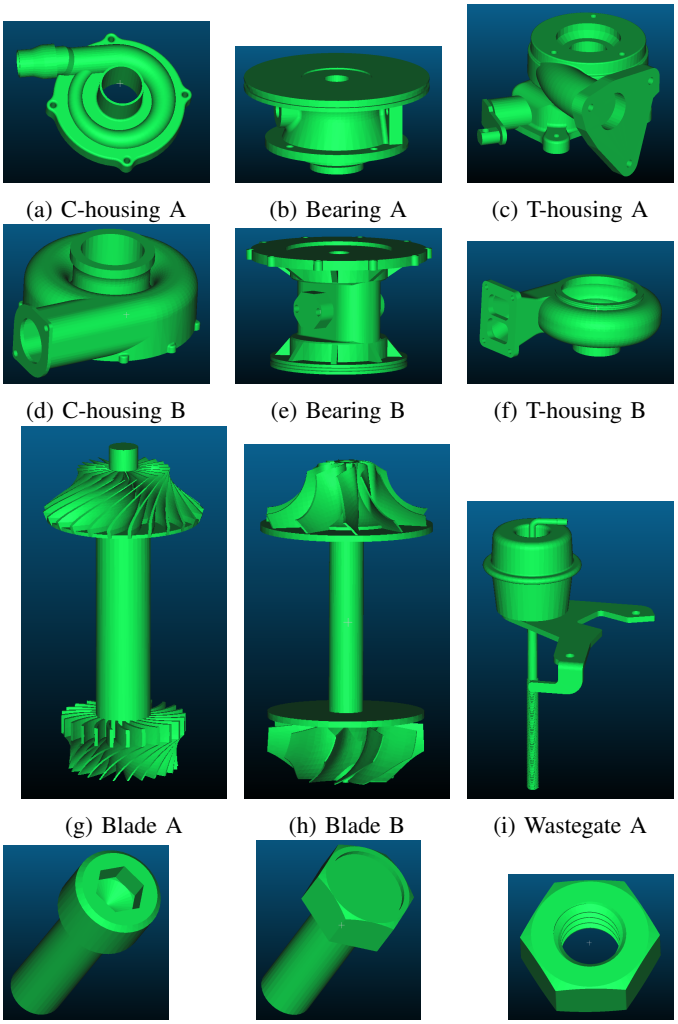


(b) Turbo charger model B

Fig. 2: The two turbochargers used to generate the point clouds

The goal of the learning procedure was to train PointNet to recognise these twelve components from point cloud models. It should be noted that the set of components contained subgroups of similar and hence potentially confusing mechanical objects such as C-housing A, C-housing B and T-housing B; Bearing A and Bearing B; Blade A and Blade B; M3 socket and M6 hex. Other objects like T-housing A, Wastegate A, and the M6 nut were clearly different from the other turbocharger components.

The training and test data sets were generated utilising a purpose-built 3D camera simulator, which mimics the depth



(a) C-housing A (b) Bearing A (c) T-housing A

(d) C-housing B (e) Bearing B (f) T-housing B

(g) Blade A (h) Blade B (i) Wastegate A

(j) M3 Socket (k) M6 Hex (l) M6 Nut

Fig. 3: The twelve mechanical parts used in experiments

capturing function of a Kinect RGB-D camera. The simulator was based on Blensor (Gschwandtner et al., 2011), and expanded to allow multiple camera capture of scenes.

The simulator places the CAD model of a mechanical part in an initial fixed position at the origin of world coordinate, and then applies a rigid transformation to rotate it into the desired pose. For most of the mechanical parts, the simulator places the virtual camera at a distance of 500mm from the origin of the world coordinate frame. For the smallest items (M3 socket, M6 hex, and M6 nut), the distance between the camera and the object was set closer (200mm) in order to obtain enough points. These scanning distances are deemed a reasonable approximation for real cameras like Kinect One and Intel Realsense.

To simulate depth sensing, the simulator emits 'light' from the matrix of pixels on the virtual camera, and records the coordinate (x,y,z) of the point where this light hits the surface of the object. In a real application, the light would be reflected back from this point to the camera sensor, and the (x,y,z) coordinates would be inferred from the light time-of-flight. The generated raw models of the mechanical parts are expressed in millimetres as unit of measurement, and reflect the real size of the objects.

Before being fed to the PointNet, the point cloud models were normalised. Normalisation rescales each object model within a bounding box of size 1 centred at the origin and parallel to the $X - Y - Z$ coordinate axes of the world frame. It should be noted that the purpose of the normalisation procedure is to 'crop' empty space out of the 3D scene, rather than to resize the object model to a standard length, width, and depth. The actual size of the object depends in fact from its orientation. For example, the size of a normalised cuboid would be the largest if its longest side was aligned to one of the coordinate axes, and smallest if the diagonal of its longest side was aligned to one of the coordinate axes.

The simulator can be used repeatedly to reproduce the effect of multiple takes on an object, in order to obtain a full view from all sides. The software allows to define the distribution of the cameras around the object. In the experiments, the point cloud models were created merging the simulated scans of three cameras from different angles. This setting simulated a realistic industrial scenario but implied a partial view of the mechanical parts. For each part, the view was determined by the orientation of the part respect to the three simulated cameras. To investigate the accuracy loss due to the limited number of cameras, point cloud models were also created using twelve virtual cameras. In this latter ideal case, a full view of the object was obtained. On average, each model contained about 100,000 points.

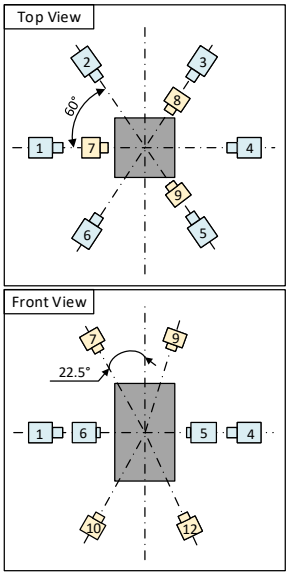


Fig. 4: Layout of the twelve cameras

The layout of the twelve virtual cameras is shown in Figure 4. Firstly, the six cameras in light blue (1-6) were placed on the X-Y plane around the object (placed at the origin). The cameras were 500mm (200mm if the object was the M3 socket, M6 hex, and M6 nut) from the origin, at angles of 60 degrees one from the other. Three additional co-planar cameras (7-9, in light yellow in the figure) were placed above

the object, 120 degrees from each other on a plane parallel to the X-Y plane, inclined 22.5 degrees respect to the Z axis and 500mm (200mm) far from the origin. The last three cameras (10-12, in light yellow in the figure) were placed below the object, symmetrically to cameras 7-9. Full view models (ideal case) of the object used all the twelve cameras, whilst partial view models (realistic scenario) used only cameras 7-9.

PointNet handles clouds of a user-defined number of points. In a number of studies (Wu et al., 2015; Qi, Su, et al., 2017), it was configured to take the 1024 points of the elements of the ModelNet40 benchmark set. Although it can be configured to process larger point clouds, the increase in computational effort impacts on PointNet learning and recall speed. For this reason, a random sampling procedure was used to pick only a fixed number of points from the clouds generated by the depth-camera simulator. Preliminary tests suggested that the generated point clouds could be undersampled by a factor 100 without loss of recognition accuracy from PointNet. Therefore, the learning trials were carried out using 1000 sampled points. One additional test was carried out to verify if halving the downsampling factor (2000 sampled points) the classification accuracy improved.

From each CAD model of mechanical part, 300 point clouds were generated by randomly rotating the CAD model, and extracting the points using the camera simulator. Of these extracted point clouds, 200 were used for training and 100 for testing purposes. In total, each training set was composed of $200\ models \times 12\ objects = 2400\ examples$, and each test set of $100\ models \times 12\ objects = 1200\ examples$. From these initial *clean* training and test sets, 10 further *error* sets were generated perturbing the position of each point.

Sensor imprecision was introduced after the normalisation of the models, and was measured as a percentage of the length of the bounding box. That is, a 1% error level meant that each x,y,z coordinate of each point was perturbed of an amount randomly sampled with uniform probability within the interval $I = [-0.005, +0.005] \in \mathbb{R}$ (the size of the bounding box is 1).

D. The Experimental Tests

The following sets of experiments were performed. The difference in learning accuracy between full and partial view of the objects was first evaluated using the clean set of models. For each view, training and validation were performed sampling 1000 points from the model. Additionally, one test was performed doubling the number of sampled points for the partial view case. In total, this set of experiments included three cases:

- full view and 1000 input points
- partial view and 1000 input points
- partial view and 2000 input points

For each case, 10 independent learning trials were performed and the results averaged. At the end of this first set of experiments, the sampling of the point clouds was fixed.

The second set of experiments includes the following cases:

- training set with zero error (clean set) and validation set with zero error
- training set with zero error and validation set with error increased from 1 to 10% in unitary steps
- training set with 5% error and validation set with error increased from 1 to 10% in unitary steps

For each case, 10 independent learning trials were performed and the results averaged.

It should be noted that the clean training and test sets contain respectively 200 and 100 different partial views (depending on the orientation of the object) of the same mechanical part. They are useful as a baseline for learning and to assess the impact of the incompleteness of object models, and any confusion that may arise in the learning of similar objects. The *error* data sets allow to evaluate the impact of the imprecision of the sensors on PointNet recognition accuracy.

Finally, one last set of experiments was carried out to test the possibility of implementing one classifier for each of the 12 mechanical parts, either to look for specific items, or to set up an ensemble of classifiers (Rokach, 2010). In this case, each of twelve independent PointNets was trained to recognise only one of the twelve parts. The training and test set of examples were the same used in the previous experiments. However, each example was now labelled as either a positive instance of the sought part (e.g. C-housing A), or a negative instance (any of the other 11 components). In this case, the main difficulty for the classifier was to learn from a highly imbalanced training set including 200 positive instances and 2200 negative instances. The test sets were composed of 100 positive instances and 1100 negative instances of the sought class. The third set of experiments includes the following cases repeated for each of the twelve classifiers:

- training set with zero error (clean set) and validation set with zero, 5% and 10% error

For each case, 10 independent learning trials were performed and the results averaged. The tests were run using the original (Qi, Su, et al., 2017) PointNet source code made available by the creators on Github (Qi, 2017), keeping the DNN architecture unchanged. PointNet was trained using ADAM optimiser (Kingma & Ba, 2014). ADAM was parameterised as shown in Table I, broadly following the settings used by Qi, Su, et al. (2017) except for the number of learning epochs (200), which was experimentally optimised. Batch normalisation (BN) was used to reduce the covariate shift during learning. The batch size (100) was optimised by trial and error.

IV. Results

This section presents the results of the experimental tests described in Section III-D.

A. Partial View and Sampling

The accuracy results of the first set of experiments are shown in Figure 5. The box plots visualise the five-number summary (sample minimum, lower quartile, median, upper quartile, and maximum) of 10 independent learning trials. The learning curve of one sample run of the PointNet training

Parameters	Value
learning epoch	200
batch size	100
learning rate	0.0001
learning rate decay rate	0.7
learning rate decay step	200,000
minimum learning rate	0.00001
BN initial momentum	0.5
BN momentum decay rate	0.5
BN momentum decay step	200,000
BN minimum momentum	0.01

TABLE I: Parameterisation of the learning algorithm

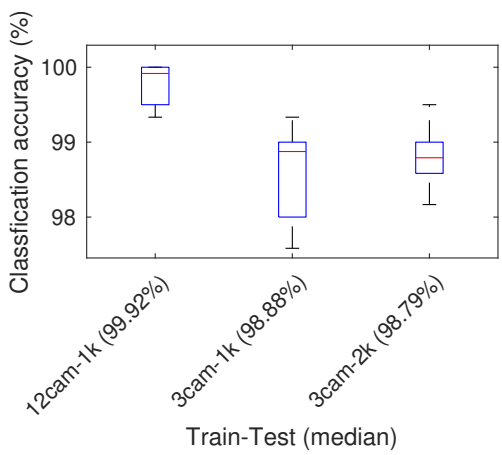


Fig. 5: Classification results for full view (12cam-1K) and partial view (3cam-1K) of the parts using 1,000 sampled points, and 2000 (3cam-2K) sampled points

procedure is shown in Figure 6. The plot shows the classification accuracy approaching 100% and stabilising after approximately 100 learning cycles.

Pair-wise two-tailed Mann-Whitney tests were performed to assess the statistical significance of the differences among the results obtained. The significance level was set to 5%. The p-values suggests a there is a significant difference ($p = 0.0002$) between the accuracy results obtained using the full view and those obtained using the partial view. However, it should be noted that the magnitude of the difference between full and partial view object identification is very modest (1%), and that in both cases the accuracy was very close to 100%.

The Mann-Whitney tests also indicated that 1000 sampled elements (1% circa of the total points) from the point cloud model was enough to achieve very high recognition accuracy, and that doubling the number of sampled points had no effect on the accuracy of the classifier ($p = 0.5708$). Consequently, it was decided to use 1000 sampled points for the remainder of the experiments.

B. Tolerance to Noise

The accuracy results of the second set of experiments are shown using the box plots in Figure 7 and Figure 8.

When the PointNet was trained on clean (zero error) scenes, the generalisation ability began to deteriorate as the error level

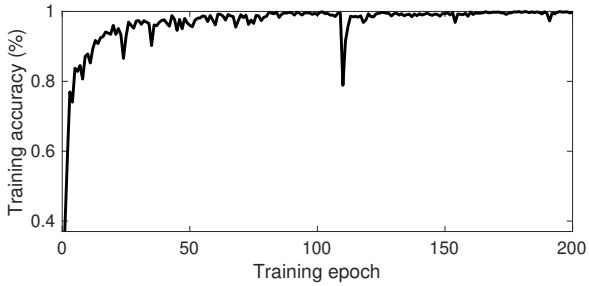


Fig. 6: Training accuracy of one of the network trained by partial view clean data set with 1000 sampled points

Test set Noise level	Training set Clean	(P-value)	5%-Noise
0%	**98.88**	0.0002	91.92
1%	**98.67**	0.0002	93.46
2%	**98.54**	0.0002	95.83
3%	98.25	0.0284	97.58
4%	97.25	0.0343	98.42
5%	94.08	0.0009	**98.63**
6%	87.00	0.0002	**98.29**
7%	76.58	0.0002	**97.25**
8%	63.58	0.0002	**93.67**
9%	52.92	0.0002	**88.04**
10%	52.92	0.0002	**79.63**

TABLE II: The p-values (pairwise Mann-Whitney tests) indicate significant differences in accuracy between classifiers trained on clean scenes and classifiers trained on scenes with 5% error level. If statistically significantly superior, accuracy values are in bold.

approached 5%. For more severe levels of noise (7% and above), the performance of the classifier fell below acceptable standards.

When a 5% error was added to the trained scenes, accuracies close to or above 90% were obtained on the test set for nearly all error levels (0-9%). Table II compares the results visually shown in Figure 7 and Figure 8, and includes the p-levels of pairwise two-tailed Mann-Whitney tests. The table clearly shows that, if sensor inaccuracy is expected, training the PointNet on scenes including a reasonable amount of error significantly improves the accuracy in the recall phase.

C. Part Specific Classifiers

The accuracy results of the third set of experiments are shown in Tables III to V. Given the highly imbalanced distribution of the class instances in the test set (Section III-D), the overall classification accuracy alone would not be a fair measure of the performance of the PointNets, as it would over-represent the accuracy on the largest class. Hence, Tables III to V report for each mechanical part (i.e. each PointNet classifier) the average (statistical mean) of true positives (TP), false positives (FP), true negatives (TN), false negatives (FN), and overall classification accuracy (A) respectively for a zero, 5%, and 10% error in the test set.

The results show how the overall accuracy could be improved by training different classifiers to recognise single types of parts. However, the improvements are limited to a better ability to recognise true negatives, whilst the performances on

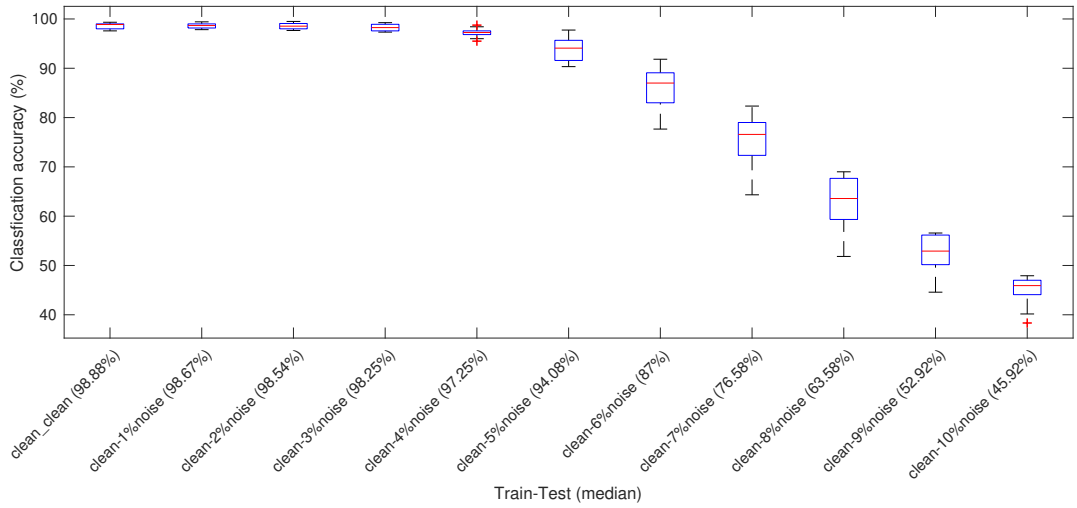


Fig. 7: Results on test sets of different error level, when the PointNet is trained using clean (zero noise) scenes

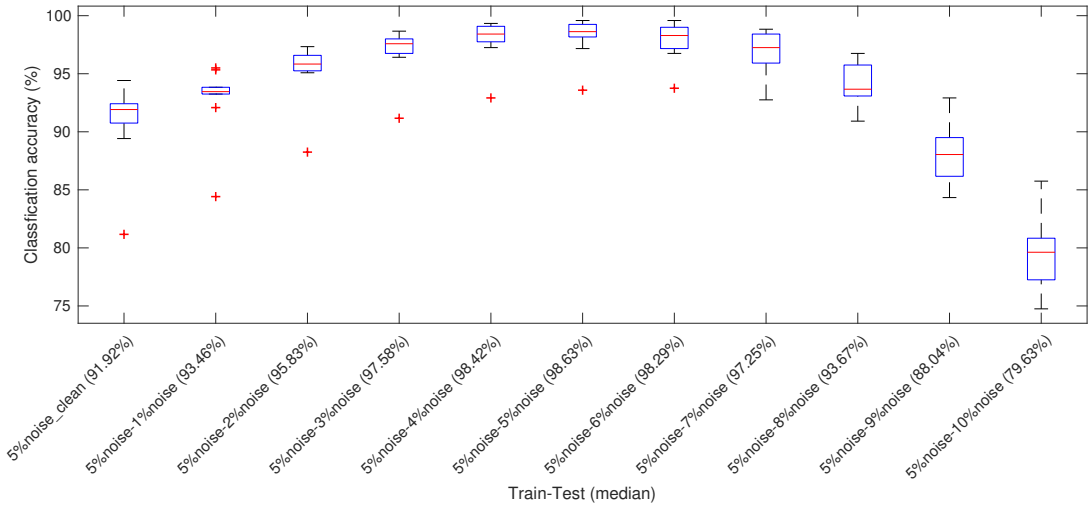


Fig. 8: Results on test sets of different error level, when the PointNet is trained using scenes with 5% error level

true positives still deteriorates with error. In particular Table V (most severe error level) shows that sensor imprecision affects the performance of PointNet on the 12 mechanical parts very unevenly, with some parts still recognised with good accuracy (Bearing B) and other almost never correctly identified (M6 Nut).

V. Discussion

This study aimed to assessing the ability of PointNet to recognise complex mechanical parts from point cloud models. In general, PointNet achieved very good accuracy results when trained on partial views to the parts. This ability of recognising objects from partial information suggests that PointNet may be able also to recognise incomplete objects, as it is sometimes needed in remanufacturing. To carry out successfully the recognition task, PointNet needed only a small fraction of randomly sampled points.

	Breakdown of recognition accuracy				
	TP	FP	TN	FN	A (%)
Compressor Housing A	99.7	17.3	1082.7	0.3	98.53
Compressor Housing B	94.4	60.4	1039.6	5.6	94.50
Bearing A	100.0	10.0	1090.0	0.0	99.17
Bearing B	99.9	0.5	1099.5	1.0	99.88
Turbine Housing A	97.2	11.6	1088.4	2.8	98.80
Turbine Housing B	94.3	61.7	1038.3	5.7	94.38
Blade A	99.5	7.1	1092.9	0.5	99.37
Blade B	99.3	4.3	1095.7	0.7	99.58
Wastegate	99.9	2.4	1097.6	0.1	99.79
M6 Hex	97.9	13.4	1086.6	2.1	98.71
M6 Nut	100.0	3.2	1096.8	0.0	99.73
M3 Socket	99.9	6.7	1093.3	0.1	99.43

TABLE III: Classifier performance: training set with zero error, test set with zero error

The presence of error in the test set affected the recognition accuracy of PointNet. Until a realistic 5% error, the accuracy of PointNet was satisfactory, namely close to or above 90%. In

	Breakdown of recognition accuracy				
	TP	FP	TN	FN	A (%)
Compressor Housing A	92.6	13.6	1086.4	7.4	98.25
Compressor Housing B	98.2	175.6	924.4	1.8	85.22
Bearing A	97.9	23.0	1077.0	2.1	97.91
Bearing B	100.0	2.3	1097.7	0.0	99.81
Turbine Housing A	96.8	80.9	1019.1	3.2	92.99
Turbine Housing B	66.9	94.8	1005.2	33.1	89.34
Blade A	90.9	1.5	1098.5	9.1	99.12
Blade B	90.2	0.1	1099.9	9.8	99.18
Wastegate	96.9	0.3	1099.7	3.1	99.72
M6 Hex	78.8	67.0	1033.0	21.2	92.65
M6 Nut	87.6	0.6	1099.4	12.4	98.92
M3 Socket	87.7	0.0	1100.0	12.3	98.98

TABLE IV: Classifier performance: training set with zero error, test set with 5% error

	Breakdown of recognition accuracy				
	TP	FP	TN	FN	A (%)
Compressor Housing A	24.9	9.1	1090.9	7.51	98.53
Compressor Housing B	86.7	284.3	815.7	13.3	75.20
Bearing A	74.3	61.2	1038.8	25.7	92.76
Bearing B	99.9	10.1	1089.9	0.1	99.15
Turbine Housing A	90.6	381.1	718.9	9.4	67.46
Turbine Housing B	8.4	250.3	849.7	91.6	71.51
Blade A	21.2	0.1	1099.9	78.8	93.43
Blade B	15.5	0.0	1100.0	84.5	92.96
Wastegate	60.9	0.0	1100.0	39.1	96.74
M6 Hex	8.8	96.3	1003.7	91.2	84.38
M6 Nut	5.3	0.0	1100.0	94.7	92.11
M3 Socket	11.1	0.0	1100.0	88.9	92.59

TABLE V: Classifier performance: training set with zero error, test set with 10% error

a factory setting, camera quality, lighting, and good denoising software will determine the error level. For error more severe than 5%, the tests performed in this study showed that the performance of PointNet can be significantly improved simulating sensor error in the training set of examples.

The tests also showed that high recognition accuracy can be achieved by training one PointNet for each mechanical part. This would be a suitable solution if one particular kind of component had to be picked up from a mixed set. Overall, the accuracy results obtained by the part-specific classifiers were similar to the results obtained by one single classifier for all parts.

Further work should investigate the possibility of setting up a recognition system based on an ensemble of classifiers. Although the tests gave no indication that an ensemble of classifiers would perform significantly better than one single classifier, an ensemble of classifiers would be easier to retrain to include new parts or remove old ones. That is, it may be enough to train only one new classifier to recognise the new part, or remove one classifier if a part is not produced anymore, instead of retraining the whole system. Part-specific classifiers showed also an overall tolerance to error, although often the high accuracy regarded the recognition of negative instead of positive instances of the sought object.

VI. Conclusion

The acquisition of data for machine learning applications is often tedious and time-consuming. This study assessed the feasibility of generating large data sets of point cloud models of mechanical parts, and use them to train a PointNet DNN to recognise previously unseen examples. All training and test scenes consisted of partial views of one of a set of twelve mechanical parts. The partial views were created using a purpose-built Kinect RGB-D camera simulator, where different levels of sensor imprecision could be simulated.

The PointNet was able to recognise the mechanical parts with high accuracy, although demonstrated a less-than expected tolerance to sensor error. However, for objects of size larger than 100mm, and camera accuracy better than +/-2mm, the classifier should perform reasonably well. Furthermore, it was found that adding some error to the training set of examples significantly improved the tolerance of the classifier to sensor error.

This study assessed also the possibility of training PointNet to recognise only one of the twelve mechanical parts. Although the accuracy results were similar to those achieved using one overall classifier, part-specific classifiers could be used to create an ensemble of classifiers with gains in modularity and reconfigurability. Further work is needed to verify this hypothesis.

References

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *European conference on computer vision* (pp. 404–417).

Bdiwi, M., Rashid, A., & Putz, M. (2016). Autonomous disassembly of electric vehicle motors based on robot cognition. In *2016 ieee international conference on robotics and automation (icra)* (pp. 2500–2505).

Besl, P. J., & McKay, N. D. (1992). Method for registration of 3-d shapes. In *Sensor fusion iv: Control paradigms and data structures* (Vol. 1611, pp. 586–607).

Carlevaris-Bianco, N., & Eustice, R. M. (2014). Learning visual feature descriptors for dynamic lighting conditions. In *2014 ieee/rsj international conference on intelligent robots and systems* (pp. 2769–2776). doi: 10.1109/IROS.2014.6942941

Castellani, M., & Rowlands, H. (2008). Evolutionary feature selection applied to artificial neural networks for wood-veneer classification. *International Journal of Production Research*, *46*(11), 3085–3105.

Chua, C. S., & Jarvis, R. (1997). Point signatures: A new representation for 3d object recognition. *International Journal of Computer Vision*, *25*(1), 63–85.

Dai, G. (2019). *3D Shape Descriptor Learning* (Unpublished doctoral dissertation). New York University Tandon School of Engineering.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition* (pp. 248–255).

Engelmann, F., Kontogianni, T., Hermans, A., & Leibe, B. (2017). Exploring spatial context for 3d semantic segmentation of point clouds. In *Proceedings of the ieee international conference on computer vision* (pp. 716–724).

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., & Garcia-Rodriguez, J. (2017). A review on

deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*.

Gschwandtner, M., Kwitt, R., Uhl, A., & Pree, W. (2011). Blensor: Blender sensor simulation toolbox. In *International symposium on visual computing* (pp. 199–208).

Guerrero, P., Kleiman, Y., Ovsjanikov, M., & Mitra, N. J. (2018). Pcpnet learning local shape properties from raw point clouds. In *Computer graphics forum* (Vol. 37, pp. 75–85).

Hussmann, S., Ringbeck, T., & Hagebeuker, B. (2008). A performance review of 3d tof vision systems in comparison to stereo vision systems. In *Stereo vision.* InTechOpen.

Ijomah, W. (2002). *A model-based definition of the generic remanufacturing business process* (Unpublished doctoral dissertation). University of Plymouth.

Jaderberg, M., & Simonyan, Z. A. K. K., Karen. (2015). Spatial transformer networks. In *Advances in neural information processing systems* (pp. 2017–2025).

Johnson, M. R., & McCarthy, I. P. (2014). Product recovery decisions within the context of extended producer responsibility. *Journal of Engineering and Technology Management*, *34*, 9–28.

Kin, S. T. M., Ong, S., & Nee, A. (2014). Remanufacturing process planning. *Procedia Cirp*, *15*, 189–194.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krasin, I., Duerig, T., Alldrin, N., Ferrari, V., Abu-El-Haija, S., Kuznetsova, A., … Veit, A. (2017). Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, *2*, 3.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Krüger, J., Lehr, J., Schlüter, M., & Bischoff, N. (2019). Deep learning for part identification based on inherent features. *CIRP Annals*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., … Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755).

Liu, J., Zhou, Z., Pham, D. T., Xu, W., Ji, C., & Liu, Q. (2018). Robotic disassembly sequence planning using enhanced discrete bees algorithm in remanufacturing. *International Journal of Production Research*, *56*(9), 3134–3151.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *iccv* (Vol. 99, pp. 1150–1157).

Pham, D., Packianather, M., & Afify, A. (2007). Artificial neural networks. In *Computational intelligence* (pp. 67–92). Springer.

Qi, C. R. (2017). *Pointnet.* Retrieved 16.02.2020, from `https://github.com/charlesq34/pointnet`

Qi, C. R., Liu, W., Wu, C., Su, H., & Guibas, L. J. (2018). Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 918–927).

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 652–660).

Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems* (pp. 5099–5108).

Quan, S., Ma, J., Hu, F., Fang, B., & Ma, T. (2018). Local voxelized structure for 3d binary feature representation and robust registration of point clouds from low-cost sensors. *Information Sciences*, *444*, 153–171.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, *33*(1-2), 1–39.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. R. (2011). Orb: An efficient alternative to sift or surf. In *Iccv* (Vol. 11, p. 2).

Rusu, R. B., Blodow, N., Marton, Z. C., & Beetz, M. (2008). Aligning point cloud views using persistent feature histograms. In *2008 ieee/rsj international conference on intelligent robots and systems* (pp. 3384–3391).

Rusu, R. B., & Cousins, S. (2011). Point cloud library (pcl). In *2011 ieee international conference on robotics and automation* (pp. 1–4).

Salvi, J., Pages, J., & Batlle, J. (2004). Pattern codification strategies in structured light systems. *Pattern recognition*, *37*(4), 827–849.

Schonberger, J. L., Hardmeier, H., Sattler, T., & Pollefeys, M. (2017). Comparative evaluation of hand-crafted and learned local features. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1482–1491).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 2818–2826).

Torres, F., Gil, P., Puente, S., Pomares, J., & Aracil, R. (2004). Automatic pc disassembly for component recovery. *The international journal of advanced manufacturing technology*, *23*(1-2), 39–46.

Umeda, K., & Arai, T. (1996). Three-dimensional vision system for mechanical assembly/disassembly. *Advanced robotics*, *11*(2), 147–167.

Vongbunyong, S., Kara, S., & Pagnucco, M. (2013a). Application of cognitive robotics in disassembly of products. *CIRP Annals*, *62*(1), 31–34.

Vongbunyong, S., Kara, S., & Pagnucco, M. (2013b). Basic behaviour control of the vision-based cognitive robotic disassembly automation. *Assembly Automation*, *33*(1), 38–56.

Vongbunyong, S., Kara, S., & Pagnucco, M. (2015). Learning and revision in cognitive robotics disassembly automation. *Robotics and computer-integrated manufacturing*, *34*, 79–94.

Wegener, K., Andrew, S., Raatz, A., Dröder, K., & Herrmann, C. (2014). Disassembly of electric vehicle batteries using the example of the audi q5 hybrid system. *Procedia CIRP*, *23*, 155–160.

Wegener, K., Chen, W. H., Dietrich, F., Dröder, K., & Kara, S. (2015). Robot assisted disassembly for the recycling of electric vehicle batteries. *Procedia Cirp*, *29*, 716–721.

Weimer, D., Scholz-Reiter, B., & Shpitalni, M. (2016). Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. *CIRP Annals*, *65*(1), 417–420.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1912–1920).

Xing, X., Cai, Y., Lu, T., Cai, S., Yang, Y., & Wen, D. (2018). 3DTNet: Learning Local Features Using 2D and 3D Cues. In *2018 international conference on 3d vision (3dv)* (pp. 435–443).

Yang, S., Xu, W., Liu, Z., Zhou, Z., & Pham, D. T. (2018). Multi-source vision perception for human-robot collaboration in manufacturing. In *2018 ieee 15th international conference on networking, sensing and control (icnsc)* (pp. 1–6).

Zagoruyko, S., & Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4353–4361).

Zhou, Y., & Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 4490–4499).