3D Point Cloud Segmentation and Automation for Railway Catenary Arches

PLAMEN BOZOV, University of Twente, The Netherlands

Point Clouds are increasingly utilised for purposes requiring object modelling and the creation of a 3D representation. Using LiDar sensors, the Dutch company "Strukton" is scanning railroad rails in order to simplify their inspection process with the aim to move to digitalization. A previous research was able to create a data pipeline that accepts a point cloud as input and produces a 3D CAD model as output. Using this already-implemented data pipeline, this project aims to optimise its procedures, rework it so that it may be utilised for the whole catenary arches of railways, and ultimately automate it completely. The pipeline works with two main processes - segmentation and object retrieval. This research focuses on modifying the RANSAC sampling and fine alignment processes from [21] and explores techniques for CAD placement to achieve a fully automated data pipeline.

Additional Key Words and Phrases: Deep learning, point cloud, visual recognition, segmentation, data pipeline, automation.

1 INTRODUCTION

To achieve a more efficient inspection process, NS collaborates closely with Strukton (a company specialising in the development of railway infrastructure and maintenance techniques) to develop new solutions for identifying the parts that lie on the railway arches, their condition, and whether they require maintenance, so that the process can be performed by sensors and a point cloud machine learning algorithm instead of railway personnel. A study team from "Saxion University of Applied Sciences" has been tasked with determining if this work can be accomplished and if such a system can be installed. The railway's arches are scanned using a LiDar sensor. The scanned area is first converted to a 3D point cloud and then to a 3D model. The point cloud is segmented via the deep learning technique PointNet++. As this has become the core for point cloud segmentation and is commonly used for such jobs [11], it still possesses drawbacks, such as a low number of points that can be processed [6], which impacts the model size produced [20]. In the research that will be undertaken, it will be explored how the algorithm's performance can be enhanced by examining different point cloud alignment strategies for automating the data pipeline flow. The objective of the research is to combine previously created components and available data to create a completely automated data pipeline that can be used to generate CAD models without manual input. To reach that goal, the current placement strategy with using the open source tool CloudCompare will be replaced with an algorithm that performs Voxel modeling with the support of Python libraries for 3D modeling and visualization. The data pipeline consists of two key components, the first of which is the

TScIT 37, July 8, 2022, Enschede, The Netherlands
© 2022 University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

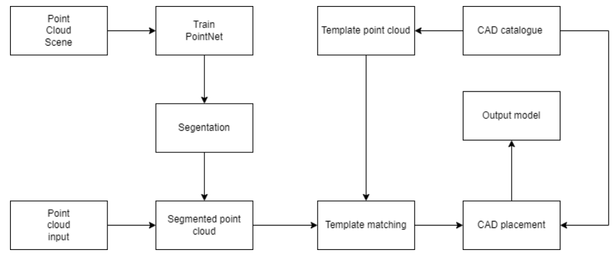

Fig. 1. Brief overview of resulting data pipeline

point cloud segmentation, which requires the employment of a machine learning algorithm (the aforementioned PointNet algorithm) to complete. Following this, the points are pre-processed, and the resulting point cloud is matched with a corresponding template, allowing CAD placement and the production of a 3D model.

The above-mentioned leads to the following research question:

RQ: How to create and use segmentation models for fully automated data pipeline for transforming 3D point clouds of railway catenary arches to scenes of digital image 3D models. The subquestions that the upcoming research will aim to answer are

(1) How to improve the existing template matching algorithm
(2) To analyze and improve the heuristics of the already existing models.
(3) How to fully automate the data pipeline segmentation models, by improving the already existing ways and extending the proposals of the previous research papers [21].

Fig 1. depicts the real core processes of the data pipeline and their interrelationships.

In the following research paper, firstly it will be done a literature overview on already existing work that supports the implementation of the automated data pipeline. After that, the two methodology techniques used for training the PointNet machine learning algorithm and the development of the data pipeline automated components will be explored with how they support the implementation of the necessary processes. Next the data pipeline will be discussed with all the processes inside it, tried techniques for improvement and the results from their usage. Following to that will be presented the technical aspect of the data pipeline automation. In the end of the paper, the results will be discussed and future work as continuation will be suggested.

2 LITERATURE OVERVIEW

As a main technology for performing the segmentation process, the data pipeline uses the PointNet algorithm that has developed as a core for performing point cloud semantic segmentation and classification [14]. [9] Explores the applications that the PointNet++ algorithm, that is also used for this project, has to perform segmentation of point clouds with high and low density of points. As stated

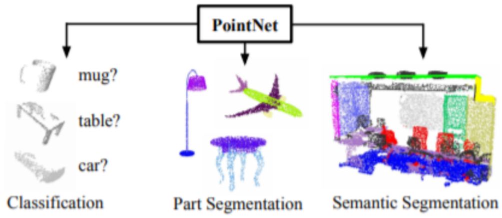

Fig. 2. Example of point cloud segmentation [8]

by [20] The PointNet algorithm can perform well in almost every type of point cloud, but it starts to struggle with objects with lower density of points. There are three objects with which reasoning can be formed for objects [8]. One is object classification, which is a process of computing a set of geometric attributes of a point cloud in order to be defined in particular objects. For example, when a point cloud is read to be able to recognize objects like chair, table and so on. The classification can be performed based on doing partial segmentation, which is defined by finding parts of the point clouds with semantic meaning to the provided data. This means performing the segmentation process directly on the smaller objects. Semantic segmentation is dividing the point cloud into regions where the points in the same region have the same properties [14]. Fig 4. and Fig 5. show a point cloud of a catenary arch before and after being run through the segmentation process.

The PointNet algorithm used for this research is developed by Bram Ton [19] with modifications to do a semantic segmentation on catenary arches. The algorithm scores high accuracy on the provided input with ninety-three to ninety-six percent, which can be considered relatively high. The algorithm was found to score lower on smaller object like insulators [21], but as suggested by [14] this can be corrected by implementing a fine alignment on those components. The algorithm performs labeling by classifying the sixteen objects on a catenary arch and performs coloring by setting a different color scheme to each classification as described by [19].

The research that is being performed is a continuation on the one performed by Zino Vieth [21] with the aim to use the base that was achieved in that iteration and to extend it to achieve a full automation of the data pipeline. What was developed was a data pipeline that produces a 3D model of the pole and the insulator of a catenary arch by performing the segmentation algorithm developed by [19] and develops a template matching algorithm that uses RANSAC sampling to produce a CAD model. Two main parts of the data pipeline were not automated during that iteration, which are the fine alignment, which was implemented, but specifically for the insulator of the catenary arch, and the CAD placement process, for which it was manually used CloudCompare. As a continuation, this research will automate those two parts of the data pipeline, as well as it will automate the flow of data between the different processes that are being executed on the point clouds.

3 RESEARCH METHODOLOGY

The research will be conducted in two stages in order to produce valid results and contribute to the work already accomplished by the

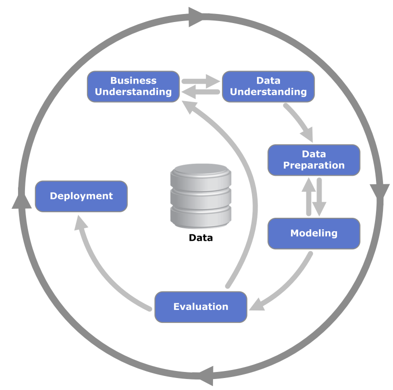

Fig. 3. The process flow of CRISP-DM, by [17]

research group at "Saxion University of Applied Sciences." CRISP-DM, which is primarily and commonly used for approaching machine learning research problems with its personalized working schema for the given subject [23] would be used as the research method. This method is also used by the other researchers in the research group and in the prior research project articles on this work; therefore, for consistency, this research will follow the same paradigm. This research methodology is used for the segmentation process of the data pipeline. For the automation will be used Design Science methodology, which will be introduced in 3.4.

The diagram in Fig 3. illustrates the execution order of CRISP-DM's as a flowchart. The subsequent subparagraphs describe which procedure and component are associated with this project.

3.1 Business understanding

The aim of the project is to construct an automated data pipeline that can take a point cloud as an input, can perform the necessary processes to modify the provided input to construct a model and in the end to output a CAD model of the provided catenary arches. The success of the project is dependent on whether in the end the pipeline can produce models without the need of manual adjustments on the point cloud and restricting the pipeline on specific arches, but it should be able to process a point cloud no meter of the provided input.

3.2 Data understanding

The provided data is stored in files with LAZ format, which is a way to compress a point cloud. There were fifteen arches provided as separate files from the mentioned format. The density of each point cloud is relatively high with four million points per catenary arch, or it can also be considered as such amount of point density per file. Every catenary arch consists of different parts that can either be isolated as a separate component or can be considered as merged to a bigger part and can not be reproduced in a separate CAD model. CAD catalog was also provided in the data, that consists of fifty-nine different models that contain different parts of the arches like the pole and the insulator etc.

This visualization in Fig 4. depicts an example data set arch before to being run by all data pipeline procedures. On the image, the points

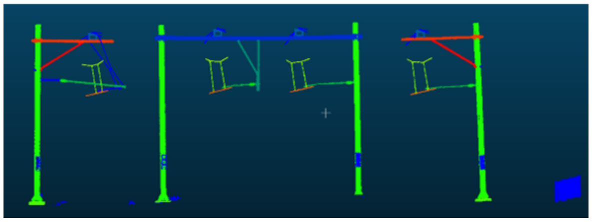

Fig. 4. Example of a catanary arch before being processed

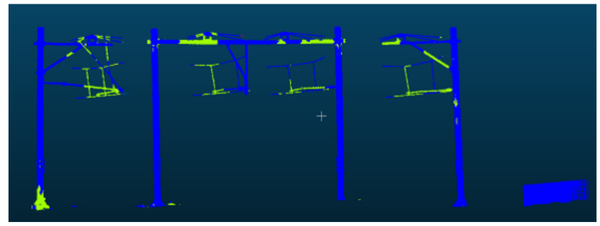

Fig. 5. Catenary arch after being modelled with the PointNet algorithm.

appear to be labelled and colored, however this is not the case, as the labeling happens after the segmentation process.

3.3 Data preparation

The data preparation happens during the segmentation process from the PointNet algorithm. Every point cloud file that goes through the segmentation algorithm is being labelled, cleaned, down-sampled and split. Because the aim of the project is to construct a data pipeline that can place a complete catenary arch, the data splitting will be skipped, because there is no need to take into account the different components that are contained.

The cleaning process is making a selection of the points and removing the ones that are not necessary to be part of the point cloud. This points in most cases are considered as outliers, because they are separated from the density regions.

The labeling process can be considered as drawing bounding boxes around the points in the point clouds. The labeling adds the properties to the points as the characteristics that define it like placement coordinates in the point cloud as well as the color properties of the points.

The down-sampling process reduces the density of the point clouds to ease the process of the processing of the data to the final stage of placing a CAD model. In the project there are two techniques that are going to be explored - RANSAC sampling which is part of the template matching algorithm of the data pipeline as well as Voxel modeling and voxel down-sampling which repositions the points to construct a finer model.

3.4 Modeling

After the data is prepared, the PointNet algorithm can be trained with the provided training data and can start the segmentation process on the actual data. In the provided Fig 5. can be seen a catenary arch after being processed with the PointCloud algorithm. There are selected two main colors for the parts of the arch, which are the stable colored in blue and the components responsible for the train movement that are colored in green.

The activities described above pertain to the data science portion of the project, specifically the training of the PointNet algorithm and the segmentation of points in the data pipeline. What follows is the remaining portion of the data pipeline. The software developed to automate it will follow a different implementation style. The subsequent procedure is Design Science Methodology [22]. The representation in Fig 6. illustrates the phases and sequence followed by the processes of the selected approach.

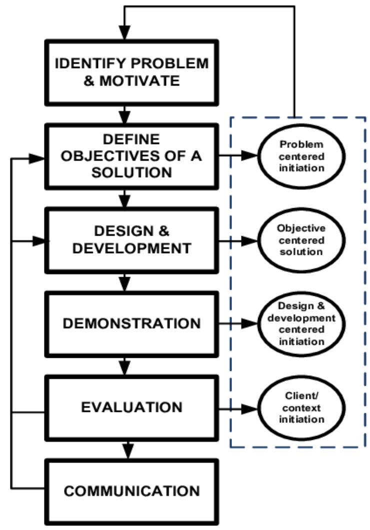

Fig. 6. Design Science methodology flow [2]

3.4.1 *Problem and Motivation.*

The primary issue is that the current data pipeline is not entirely automated. This procedure required an initial review of existing pipeline implementation. All the procedures were present, but they were designed to operate manually and independently. Only partial automation of template matching processes existed previously. The primary issue was that with each step, a new file was loaded and produced, which had to be used in the subsequent process. The fine alignment algorithm was designed to work only for the pole and the insulator of the catenary arch, which requires adopting a new one that can be able to work on the full input point cloud. The CAD placement process is being done manually, which also requires further adjustments to the existing data pipeline to be able to produce a 3D model in an automated way.

3.4.2 *Define Objectives of a Solution.*

It was determined that the answer to the challenge would be divided into two sections. The first objective was to automate the point cloud segmentation. This process already works in an automated way thanks to the developed PointNet algorithm. What needed to be done was to re-train the algorithm, because it needed to be altered to work for the whole catenary arch without taking into account the moving parts that are responsible for the movement of the train and the base of the parts. In that case, the need was only to have two main labeling parts. After then, the issue was the subsequent phase of point processing prior to CAD placement. A significant portion has already been implemented automatically, but just for one file, which required developing a script to automate the flow of files. The supplied function required modification to accept the file as a parameter. In addition, the functions had to be built so as not to set scales directly, but to be able to handle the point cloud regardless of its density and outlier points. For automating the fine alignment, two algorithms were decided to be tested - ICP and NICP, which are explained in section 5.4. To automate the CAD placement of the catenary arches, the tried solution was to implement a voxel modeling function and use the inbuilt python 3D libraries.

3.4.3 Design and development. The working algorithm was already developed in Python, so a decision was made to continue using the same programming language, because of the inbuilt machine learning properties and libraries it has. The needed python libraries were Tensorflow and numpy for running and adjusting the machine learning algorithm. Mplot3d, pyplot and IPlotlib were used for the implementation of the tested fine alignment algorithms and open3d was used for the Voxel modeling and down sampling. Also, the inbuild functionalities from the pointcloud library were used to work directly and modify the point clouds.

3.4.4 Demonstration. The demonstration section comprises validating that the algorithms are operational and that the outputs are accurate. In this section, it is determined whether the technical aspect of the implementation is effective. For the segmentation procedure, a testing algorithm was built. For further stages of the data pipeline, the obtained results were used as a benchmark.

3.4.5 Evaluation. In this section, it is determined whether the produced code meets the requirements. Here, the produced models and their conformity to the expectations of the developed algorithms are evaluated. The results from the phase can be observed in section 7.

3.4.6 Communication. This section follows the evaluation as the final section. As a result of the evaluation, it is then determined whether the created software should restart the production cycle or whether it performs correctly and the production cycle can be concluded successfully.

3.5 Evaluation

The model described in the previous sections is evaluated based on the training and after that testing the process and comparing. The evaluation is done by the score obtained using mIoU heuristics [18]. The provided algorithm scores with 93 percent accuracy after the training and testing process. The second one can also be considered as part of the modeling phase. The process is explained into details in section 4.1.

4 PROJECT REALIZATION

In the following subparagraphs, the technologies used to transform a raw point cloud into a CAD model and the strategies needed to implement the technologies for a fully functional data pipeline will be described.

4.1 PointNet++ implementation and training

The PointNet technique was provided directly by the research group and has been developed in previous rounds of the project to work directly with the provided dataset. Setting up the development environment is the first step in getting started. For the work, a server provided by the "University of Twente" is utilized. To execute the machine learning algorithm, the server must have an Nvidia graphics card because the frameworks used to train such an algorithm are specifically designed for their components. The PointNet++ technique employs TensorFlow, which is already pre-installed by default on the server being utilized. Because of its expertise in offering a platform for writing and executing AI-based algorithms, the implementation environment is Jupiter notebook. Before executing the machine learning algorithm, it is necessary to prepare the data. Using voxel down sampling of the data, this operation begins with data sampling that reduces the number of points in the point cloud [9]. This algorithm achieves the highest score for the specified job because it uses voxel grid for point selection and uniform sampling at the same time, which creates process efficiency without introducing additional noise or data loss.

As stated in the preceding section, the preparation of data via voxel down sampling is crucial for achieving acceptable results with sufficient accuracy and adequate training time. Prior to that, the data is selected. The selected data for the project is the complete dataset of 15 arches. The provided dataset is sampled using the open3d python package and stored as files in preparation for the subsequent data pipeline processes. Due to the goal of automating the data stream, the names of the stored files are vital. The PointCloud algorithm searches for files in a certain directory with a specific name and order until it receives a NullPointerException in order to load them for the training procedure. Also, essential to the data selection process are the training and testing datasets. The data for training and testing are distinct, and it was determined for the project that the set would be split 70/30 because the set is not especially large and reducing the testing set could lead to inaccuracy for both sets during training [9]. This necessitates the departure of a larger portion for testing, estimated to be three arches.

During the process of training, the output data of the two data sets are compared with their respective heuristics. The model technique obtains an accuracy between 89 and 96 percent for both data sets. In order to avoid overfitting, it is crucial to ensure that the data for each sets contain different arches (an event that produces false results if the data is specific) [9]. In this instance, there is overfitting for catenary arches, but this risk is avoided by selecting different ones for each set.

4.2 Segmentation process

The algorithm is prepared for the segmentation procedure after training. It is chosen for semantic segmentation, which necessitates the ability to reconstruct the various components from the arches. Because the data set contains a few arches, the LOOCV learning algorithm [5], which labels the points of only one arch every segmentation iteration, is used for the labelling procedure. In order to improve the accuracy of the segmentation process during the training of the algorithm, earlier research [21] devised a method

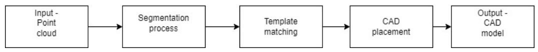

Fig. 7. Basic processes of the data pipeline

for creating a variety of arches with minor changes to aid in the training of the point cloud segmentation algorithm. In subsequent chapters, the process of the data pipeline will be described in greater detail, along with a tour of the automated components.

4.3 Evaluation of the model

The model showed good results by scoring 93 percent accuracy. The result obtained from the mIoU scale was between 0.19 to 0.27 per iteration, which can be considered as an average result, taking into account that the labeling was reduced to two color schemas. On bigger arches, the mIoU scale was increased to 0.34, which extends and confirms the previous results obtained by [21].

5 DATA PIPELINE IMPLEMENTATION

The next phase, following the preparation of the data, is to implement the processes of converting the data into a CAD model that can be placed on a plane for 3D visualisation. Before beginning the process, the point clouds are already labelled, hence the entire data preparation occurs on previously labelled data. The Fig 7. depicts the three primary processes that comprise the data pipeline. These three contain numerous subprocesses that are covered in the subsequent subchapters.

5.1 Template preparation

The initial phase of the data pipeline is to load the "Strukton" templates. The selected templates are then matched against the point clouds with which the machine learning algorithm was trained. After that, the application CloudCompare can be utilised, which provides an environment for the direct conversion of a point cloud to a 3D visual projection on a plane. Using this application, the procedure may be directly implemented to prepare the data for the process of template matching. Regarding the concept of automating the data pipeline, however, this operation can be omitted as it is unnecessary due to the subsequent step.

5.2 Voxel modeling and down sampling

For the project to model the template on the 3D plane, voxel modelling was utilised, which enabled more precise modelling due to the usage of a grid of numerous cubes that reassembles particles. In addition, the simulation approaches that voxel modelling offers cannot be reconstructed using other modelling techniques, as each point is part of a particle and the object is modelled in the center of the 3D plane by placing each point in its correct location on the grid. Using the open3d library [4] that is available for Python, voxel modelling may be performed directly. First, the bounding box must be calculated. Then, the bounding box should be extracted from the template model. To do this, we can design a cube whose edges precisely match those of the model in order to fill it in till the farthest edge. The cube is then divided into a grid of smaller cubes, each of which is 0.05 the size of the initial cube. After the production

Fig. 8. Catanary arch after the voxel process

of voxel cubes, each is centered so that it contains a single point at its center. In the end, these cubes will be assigned the labels of the points, and the voxel model will be displayed and exported as an obj file that may be utilized in subsequent data pipeline phases. By generating the voxel model, the CAD placement done with cloud compare can be directly replaced, as the output is already a visualization that can be seen in Fig 14. The algorithm produces as an output an OBJ file that can be opened, viewed and rotated by any operating system [24].

The Fig 8. shows how the points look in the voxel grid after doing the voxel modeling and voxel down sampling.

5.3 Instance segmentation

While using the given CAD catalogue from "Strukton" and generating a point cloud segmented scene, the problem arises that there cannot be a satisfactory match because each segmented scene has several components. In the previous study [21], the proposed solution to the problem was to add more points to each cluster of points and then split the clusters into distinct scenes. This procedure is carried out in the same manner as in the prior study [21] however, an additional phase is introduced in which the parameters are directly fed with values taken from the voxel modelling method so that the algorithm built by the previous researcher can be conducted automatically [12].

5.4 Fine alignment

In the project, the algorithm for fine alignment that operates on the data pipeline has restrictions for rotations of objects as well as complex scaling models for point clouds. For items such as insulators, the results are satisfactory. The algorithm that is now operational aligns each point's three dimensions by raising the shrinking values. If improved values are discovered, the method is repeated until no further improvements can be found. By doing so, the 3D positioning of the item is altered, and the points are adjusted to generate a better placement of the model for the subsequent data pipeline phase [7].However, this algorithm done by [21] works with good results only when being applied for parts of the data pipeline. Because of that, two other algorithms are ICP and NICP that showed good results when applied to the full point cloud of the catanary arches.

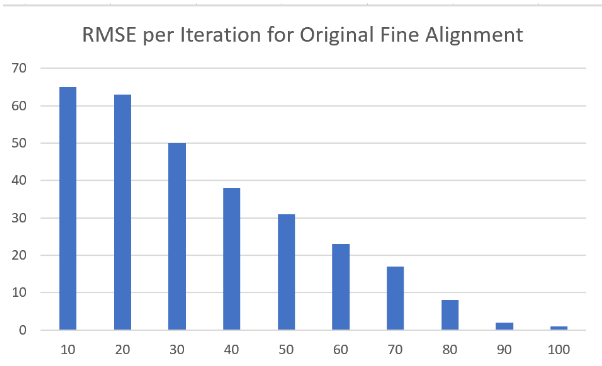

Fig. 9. Scoring for the original fine alignment algorithm

5.4.1 ICP algorithm. In order to implement the algorithm, a transformation matrix needs to be defined. Here it is directly taken from the one of the previous research. As this is a full arch actually providing a transformation matrix is not a necessary step, because the objects on the arch are not going to be moved, but as a decision it was still applied in case of rotation problems, so it can be ensured that the model will be displayed correctly on the 3D plane [15]. The idea of the algorithm is to take a parameter and to match the closest point from both the CAD catalogue and the input point cloud. The algorithm is being run again one hundred iterations to be able to be compared with the combination on Fine alignment with RANSAC sampling from [21]. Even after running the ICP algorithm, the RANSAC sampling is decided to be kept due to the reduction of the point cloud, as it enhances the performance of the algorithm when the model needs to go to CAD placement.

5.4.2 NICP algorithm. The NICP algorithm focusses on the same idea as ICP, but takes into account the surface orientation of the points on the 3D plane. A parameter is set and all the points are being moved with the same amount to be positioned correctly. This parameter can be considered as a constant [16].

5.4.3 Original Fine alignment verses ICP and NICP alignment. The three algorithm are being run for one hundred iterations, to be able to compare the results fairly. In the charts on Fig 9. Fig 10. Fig 11 can be seen how the algorithms score based on root-mean-square error per iteration [3].

5.5 RANSAC Sampling

The RANSAC sampling portion of the data pipeline is taken directly from prior research because it is already automated and can be applied to any point cloud without considering other variables. The process is a loop that calculates and stores the hash value for each point pair as it iterates across each pair. After selecting a point from the template point cloud and a random point pair value from the stored point pairs, the value of the template point is compared to the value of the point pair. This stage executes the actual template matching procedure, which is the last step prior to CAD placement. The algorithm is influenced by [13].

The visualization in Fig 12. shows the full process of the RANSAC down sampling.

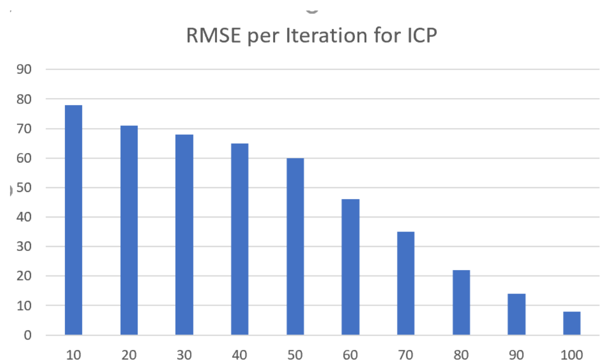

Fig. 10. Scoring for ICP algorithm

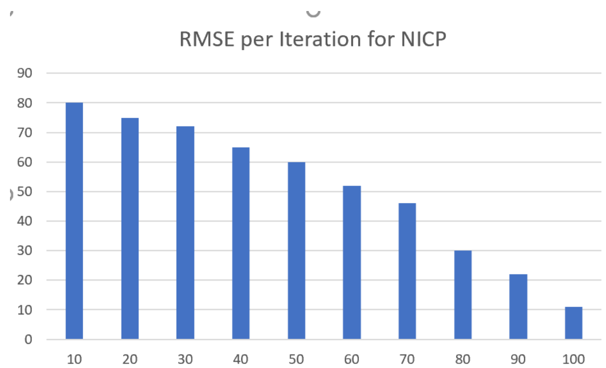

Fig. 11. Scoring for NICP algorithm

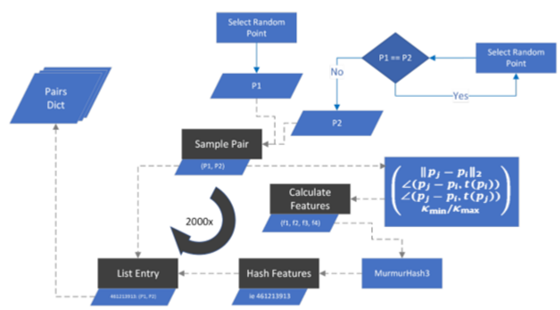

Fig. 12. RANSAC sampling pipeline taken by [21]

5.6 CAD placement

The choice was made not to automate the CAD placing process, as this final phase of the data pipeline serves solely to view the model generated by the algorithm described in the preceding paragraphs. CloudCompare [1], an open-source application designed specifically for visualizing point clouds and comparing them with segmented CAD models, was utilized for the visualization. Prior to using the instrument, the preceding actions must be completed. If not all of them, only applying the revised algorithm for voxel modelling can immediately generate a model that CloudCompare can operate with.

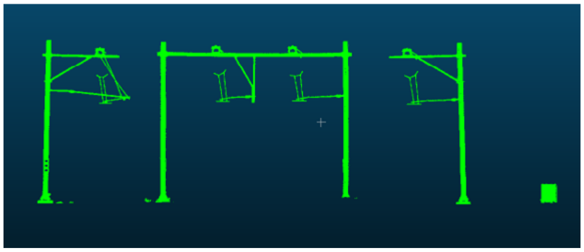

Fig. 13. Model with manual CAD placement in CloudCompare

Fig. 14. Automatic placement after using Voxel modeling

After completing all operations, the final model is saved as a binary object (obj) [1] file that can be easily imported into CloudCompare with all required features. To make the procedure work, the object files must be loaded into the application, the transformation must be applied, and finally, the correct transformation matrix must be selected and applied. The proper mesh must be chosen, and this procedure must be performed manually for each instance of the data set.

Fig 13. shows how a full catenary arch looks after doing a manual CAD placement in CloudCompare.

6 DATA PIPELINE AUTOMATION

In earlier research [21], members of the research group conducted trials to achieve the automation of the data flow described in the preceding chapters. The primary objective of this research was to fully automate the entire process, which was technically accomplished; however, certain aspects were deemed infeasible due to the necessity for an external application that required particular input and manual operations to obtain the final CAD placement. First, a point cloud was generated to train the machine learning model. The process was automated by itself, however other automations are required for this process and will be discussed. Following the production of the point cloud, the PointNet++ algorithm was trained and adapted to fit the project's objectives, which needed testing on the entire arches' components. For this portion and the remainder of the algorithm, it was determined that the entire procedure is a series of activities required to finish the algorithm. Initially, it was necessary to run the tf ops files in a special manner to generate the files required by the training script for the algorithm. These files are required to specify the correct settings for the Tensorflow plugin utilised by the hardware in order to tailor the algorithm to the laz files containing the catenary arches for the scanned train lines. After the generation, the train algorithm can be executed and trained so that it can be used in the segmentation procedure. To automate this procedure, a Python script was developed that sequentially performs the instructions required to construct the hdf5 [5] point cloud file and the files required to set the tensorflow parameters. Following this, the training algorithm is initiated [10]. The training of the model is only required once during the entire CAD placement procedure; hence, the script requires a flag to be set in order to execute the algorithm training section [10]. If it is not set, that process will be skipped and the program will go to the next section. The only automation for the process is the script itself, as the subprocesses of file generation are automated by themselves. The majority of the automation of the data pipeline occurs in the data processing portion of the flow, as here is when the majority of template matching alterations must be made. In the prior study [21], the object for CAD placement was prepared using a template matching algorithm that performed the preparation, template matrix, alignment, and final matching. This procedure could only be executed on a single data file at a time and was designed just for portions of the arches. The automation consisted of reworking the entire algorithm into one that can accept multiple files as parameters. The template matching method directly accepts a LAZ file as input, executes the procedures described in Section 5, and generates a binary object file that can be loaded directly into CloudeCompare for the CAD placement process described in Section 5.5, or directly can be taken as an output from the Voxel modeling algorithm in section 5.2. The algorithm was influenced by [8]. The most significant addition was searching for LAZ files, directly loading each one, applying the algorithm, and exporting it with a unique name to a directory from which the next steps can be taken. The procedure does not require the manual setting of boundaries for the voxel object, as in template matching, but instead locates the item's center, sets the most distant point as the boundary to the main border, and applies the processes outlined in section 6.2. It can operate on any laz file in the dataset, thus no predefined parameters are required. Ultimately, it was proposed that the CAD positioning be performed using the free tool CloudCompare. This process phase may also be automated. Due to their support for the visualization of a point cloud, libraries such as PPTK or open3d can be utilized following the preprocessing of the data. In this scenario, the prior step of exporting an object file can be ignored, but the object can be directly applied to the pptk library or taken to open3d. Open3d offers superior visualization capabilities, but also necessitates more space and processing power from the system executing the procedure. In the provided Fig 14 can be seen the result of the automated CAD placement that happens with the help of Voxel modeling.

7 DISCUSSION

Significant contributions had already been made to the point cloud project for "Strukton", which served as the foundation for the continuation of this project and the execution of the already employed algorithms. The initial plan for the project was to fully automate the data pipeline procedures. With the progress made, it was determined that the proposal for complete automation is viable, but the visual quality of the created CAD model will decrease. The application of voxel modelling yields favorable results for data pre-processing, since it enables the direct omission of steps such as instance segmentation and the CAD palcement technique used in the previous research iteration [21]. When used to a tool such as CloudCompare, the resulting visual item is distinct and meets visual expectations. Using the Python Visualization Library, the end output is once again clear, but it cannot be termed a CAD placed model because the visualization is not as clean and refined as it ought to be. Even though the arch can be modelled well, I was unable to locate a modelling technique that can produce an automated CAD model without the usage of a tool for the CAD placement procedure. Because of this, it is reasonable to conclude that a fully automated data pipeline is feasible, but the created CAD model cannot be regarded a successful data pipeline using the currently available approaches. The prior research's conclusions have not altered because this project does not significantly alter the already existing algorithms; rather, it merely modifies them in order to accomplish full automation of the data flow. The old fine alignment algorithm with RANSAC sampling scored an average of 80 percent overfitting for the components on which it was tested, however when tried on the full catenary arch, it scored 65 percent, which was considered low. When applying the proposed ICP algorithm, the score was increased to 78 percent and for the NICP algorithm, the score was increased to the results of the initial algorithm to 80 percent again, but for the full catenary arch.

8 CONCLUSION

The above document proposes a method for the full automation of the data pipeline and describes the steps that must be taken to build a 3D CAD model. The project automated the production of a model by refining the data pipeline and modifying the processes. In addition, an entirely new process called Voxel modelling was constructed, which proposes a way to automate the CAD placement procedure and replace the operations done by the open source tool CloudCompare. What was observed is that fully replacing the CAD placement with Voxel modeling does not produce good results. As observed by the provided visualization on Fig 14, and it cannot be concluded from it that the data pipeline can be automated to produce good models like a tool such as CloudCompare. However, it can be seen that full automation is possible and with a further development of the libraries that are used for working with point clouds such as open3d, the automation of the data pipeline will be possible. The greatest accomplishment of the project was the automation that resolved the issue of files being produced between operations and then having to be manually imported for the subsequent processes. This issue was resolved by modifying the template matching method and segmentation algorithm already in use. By automating this process, the results remained comparable to the prior work on this subject, but the approach was modified and adapted to operate with all types of models, such as the entire catenary arches, rather than just parts of it. To make that possible, two possible changes for the initial fine alignment algorithm were proposed, with one scoring similar results to the initial version, but working with a big object like a full catenary arch.

9 FUTURE WORK

This project offers numerous options to enhance the data pipeline in order to achieve better outcomes and automate the process. It is necessary to work on enhancing the data pipeline by omitting certain procedures or substituting them with others because the produced concept does not yield satisfactory outcomes when fully automated. An option for achieving full automation is to fully eliminate or replace the segmentation procedure and proceed without it to the next phase. Thus, a substantial portion of the pipeline will be eliminated, and subsequent operations will be left to work with the initial point cloud, which will eliminate a substantial portion of the creating and subsequently consuming file. In the event that a digital twin is required for only a portion of the catenary arches, or if a more precise template match is required, it may be necessary to update the CAD catalogue containing all the potential components. Finding an alternative modelling technique that can be utilised to automate the data flow for the CAD positioning of the final part is one option to extend the current algorithm. It can be assumed that voxel modelling and downsampling can yield good results, but in the interim, the obtained models can be considerably enhanced by discovering an alternative modelling technique. A commitment for the future of the project would be to totally restructure the procedures so that the entire process of acquiring files in the midst of the process and reloading them can be eliminated from the data pipeline and everything can operate in a continuous stream.

10 ACKNOWLEDGEMENT

I want to specially thank Faizan Ahmed for being my supervisor and the chance to work on the project as well as Bram Ton for all the technical support I received during the project.

REFERENCES

[1] CloudCompareWiki, Nov. 2020. [Online; accessed 3. Jul. 2022].

[2] Fig. 1. Example of a Design Science Research Methodology [23], Nov. 2020. [Online; accessed 3. Jul. 2022].

[3] RMSE: Root Mean Square Error, May 2021. [Online; accessed 3. Jul. 2022].

[4] Open3D-ML — Open3D 0.15.1 documentation, Feb. 2022. [Online; accessed 3. Jul. 2022].

[5] Brownlee, J. LOOCV for Evaluating Machine Learning Algorithms. *Machine Learning Mastery* (Aug. 2020).

[6] Chen, J., Kira, Z., and Cho, Y. K. Lrgnet: Learnable region growing for class-agnostic point cloud segmentation. *IEEE Robotics and Automation Letters 6*, 2 (2021), 2799–2806.

[7] Fernández-Martínez, J. L., Tompkins, M., Mukerji, T., and Alumbaugh, D. Geometric sampling: An approach to uncertainty in high dimensional spaces. *Advances in Intelligent and Soft Computing* (2010), 247–254.

[8] Gujar, S. Pointnet - Sanket Gujar - Medium. *Medium* (Sept. 2020).

[9] Guo, Y., Wang, H., Liu, H., Liu, L., and Bennamoun, M. *Deep Learning for 3D Point Clouds: A Survey.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019.

[10] Kim, H., Yeo, C., Lee, I. D., and Mun, D. *Deep-learning-based retrieval of piping component catalogs for plant 3D CAD model reconstruction.* in Computers in Industry, 2020.

[11] Li, Y., Wen, W., Miao, T., Wu, S., Yu, Z., Wang, X., Guo, X., and Zhao, C. Automatic organ-level point cloud segmentation of maize shoots by integrating high-throughput data acquisition and deep learning. *Computers and Electronics in Agriculture 193*, 10670 (2022), 2.

[12] Poux, F., and Billen, R. Voxel-based 3d point cloud semantic segmentation: Unsupervised geometric and relationship featuring vs deep learning methods. *ISPRS International Journal of Geo-Information 8*, 5 (2019), 213.

[13] Poux, F., Mattes, C., Selman, Z., and Kobbelt, L. Automatic region-growing system for the segmentation of large point clouds. *Automation in Construction 138* (2022), 10425.

[14] Qi, C. R., Su, H., Kaichun, M., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *on Computer Vision and Pattern Recognition, Honolulu,* (USA, 2017), I. Conference, Ed., HI.

[15] Serafin, J., and Grisetti, G. Using Augmented Measurements to Improve the Convergence of ICP. In *Simulation, Modeling, and Programming for Autonomous Robots*. Springer, Cham, Switzerland, 2014, pp. 566–577.

[16] Serafin, J., and Grisetti, G. NICP: Dense normal based point cloud registration. *undefined* (2015).

[17] Sigit, K., Dewi, A. P., Windu, G., Nurmalasari, and Kadinar, N. Comparison Of Classification Methods On Sentiment Analysis Of Political Figure Electability Based On Public Comments On Online News Media Sites. *IOP Conf. Ser.: Mater. Sci. Eng. 662* (Nov. 2019), 042003.

[18] Soilán, M., Riveiro, B., Balado, J., and Arias, P. Comparison of heuristic and deep learning-based methods for ground classification from aerial point clouds. *Int. J. Digital Earth 13*, 4 (Sept. 2019), 1–20.

[19] Ton, B. . A. F. linssen, j. Tech. rep., Semantic segmentation of railway catenary arches. ISPRS Journal of Photogrammetry and Remote Sensing (Preprint), 2022.

[20] Verburg, F. M. *Exploring explainability and robustness of point cloud segmentation deep learning model by visualization.* Electrical Engineering, Mathematics and Computer Science, EEMCS, 2022.

[21] Vieth, Z. J. *POINT CLOUD CLASSIFICATION AND SEGMENTATION OF CATENARY SYSTEMS.* BSc. Creative Technology, Faculty of EEMCS, University of Twente, 2022.

[22] Wieringa, R. Design science research methods and writing. Tech. rep., Research Papers. roelw/DSM180minutes.Pdf. https://wwwhome.ewi.utwente.nl/roelw/DSM180minutes.pdf, 2016.

[23] Wirth, R., and Hipp, J. *CRISP-DM: Towards a Standard Process Model for Data Mining 10* (1999), 1.

[24] Yao, X., Guo, J., Hu, J., and Cao, Q. Using deep learning in semantic classification for point cloud data. *IEEE Access 7* (2019), 37121–37130.