CHALMERS
UNIVERSITY OF TECHNOLOGY

From image to graph: Topological representation of Wire harnesses using deep learning

Master thesis in Industrial and Materials Science

Magnus Bryntheson

Department of Industrial and Materials Science

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

From image to graph: Topological representation of Wire harnesses using deep learning

Magnus Bryntheson

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Industrial and Materials Science
Chalmers University of Technology
Gothenburg, Sweden 2025

From image to graph: Topological representation of Wire harnesses using deep learning
Magnus Bryntheson

Examiner: Björn Johansson, Department of Industrial and Materials Science
Supervisor: Hao Wang, Department of Industrial and Materials Science
Advisors: Ludwig Friborg and Patrick Andersson, Wiretronic

Master's Thesis 2025
Department of Industrial and Materials Science
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2025

From image to graph: Topological representation of Wire harnesses using deep learning
Department of Industrial and Materials Science
Chalmers University of Technology

Abstract

The ability to classify an abstract representation into an image of a wire harness is a key step in automating the wire harness assembly industries. This study examines different methods and machine learning architectures to extract visual and physical features from an image of a wire harness. To identify suitable methods for abstractly representing and classifying a wire harness, a systematic literature review was conducted. The systematic literature review was not limited to studies about wire harnesses, but expanded to studies about deformable linear objects. Following the systematic literature review, an experimental study was conducted that evaluated an existing implementation for graph representation of a wire harness and a novel method based on a You Only Live Once (YOLO) segmentation model together with a Graph Convolutional Network (GCN) to classify the graph representation against a validation file of known harnesses. The segmentation output from the YOLO network is fed into a skeletonization method that returns a graph that represents the wire harness. In order to classify the graph representation against known harness structures, a GCN along with a validation step is performed. The models are trained on a proprietary dataset and two open-source datasets. The proposed solution can be easily adapted to classify new wire harnesses.

Keywords: Computer vision, Deep learning, Wiring harness, Abstract representation, Classification, Deformable linear objects, Graph representation, Machine learning.

Acknowledgements

I would like to thank my supervisor, Hao Wang, for his valuable support and feedback during this thesis. I would also like to thank Björn Johansson who served as examiner to this master thesis guiding the work. Last but not least, I would like to thank Wiretronic, especially Ludwig Friborg and Patrick Andersson for their help throughout this thesis.

Magnus Bryntheson, Gothenburg, 06 2025

Contents

List of Figures

List of Tables

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

BDLO	Branched deformable linear objects
CV	Computer vision
DL	Deep learning
DLO	deformable linear objects
CNN	Convolutional Neural Network
D-CNN	Deep Convolutional Neural Network
GCN	Graph Convolutional Network
GNN	Graph Neural Network
ImageNet	Large open source image database with over 14 million images.
IoU	Intersection over Union
mAP	Mean average percision
ML	Machine learning
MSE	Mean squared error
NN	Neural network
RL	Reinforcement learning
YOLO	You Only Look Once

1
Introduction

1.1 Background

The manufacturing industry is highly dependent on robotic assembly to handle production needs while reducing cost, human error, and injuries [32]. However, in the automotive industry and many other fields where wiring harnesses are used, they are still manufactured by manual assembly [38]. The manual assembly of wire harness is one of the bottlenecks for production needs [38]. Although there are many challenges to fully automate the wiring harness assembly, extracting a graph representation of a wiring harness from an image is a step on the way towards automation. One of the reasons it is hard to extract a graph representation of a wring harness is due to the deformable nature of wires, more broadly called deformable linear objects (DLO) [31]. This makes the number of possible configurations almost indefinite for each harness. Rigid objects are much easier to label with traditional bounding boxes. Trying to fit a wire harness within a bounding box results in most of the bounding box consists of background 1.1. In recent years, deep learning solutions such as convolution neural networks (CNNs) have exploded in various applications [9]. Recent progress in pixel-level labeling has made it possible to capture these deformable objects much easier [20]. Being able to extract a graph representation of a wiring harness, could help automating industries that rely heavily on wiring harnesses like automotive and aviation industries. They are also applied to reduce the failure of the wiring harness assembly on the manufacturing line [20].

1.2 Problem description

The problem in this paper is, from an image of a wire harness extract a graph representation for down-the-line classification, it involves deciding on a format of how to represent a wire harness as a graph. The wire harness community suffers from data scarcity, with few annotated wiring harnesses available publicly. The deformable and branched nature of the wiring harness with possible overlapping wires makes the number of configurations infinite. Current solutions for automation of wiring harness assembly are that they often rely on artificial markers or other features to recognize a harness [31]. A computer vision model capable of extracting a graph representation only based on its visual properties is a solution needed for automation of wire harness assembly. A possible solution is semantic segmentation

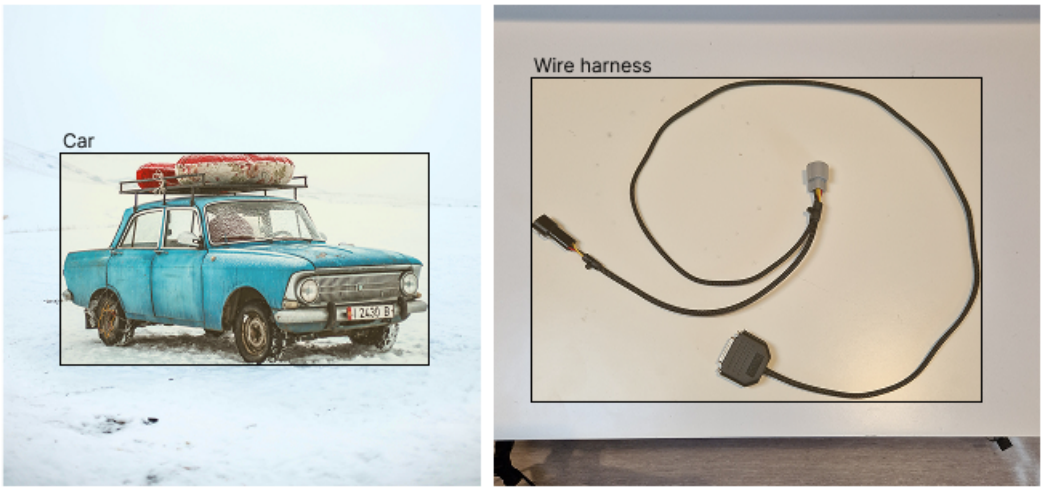


Figure 1.1: Comparison of bounding box around a rigid object versus a wire harness

that manages to extract a segmentation of the harness that can be used to extract a graph representation.

1.3 Purpose

The number of wire harnesses that are being produced is constantly growing [31]. Being able to create a method that can extract a graph representation of a wiring harness is an important step towards automation of wire harness assembly. This could later be used in applications for real-time visualization on the assembly line, assessment of harness completeness, and quality assurance. The aim of this thesis is to create a model that can extract the graph representation of a wire harness from an image of a wire harness. The model should also be able to classify the components within the wire harness, such as connectors, terminals, etc. This will be useful for classification against known harnesses and result in an informative graph representation. A recent study has been able to extract a graph representation from an image of a wiring harness [23]. [23] estimates each wire segment with keypoints to extract a graph representation. Another possible solution is to take advantage of semantic segmentation. Being able to segment the wire harness accurately will help extract additional features such as color, length, geometrical features, etc. This contributes to a more detailed graph representation.

1.4 Research Questions

This paper will investigate and answer these questions to complete the goal of classifying a graph representation of a wire harness to an image of a wire harness with computer vision. To gain knowledge of previous work and what different visual cues such as wire, connectors, etc. that a wire harness exists of **RQ1** will be investigated. As mentioned above, [23] manages to extract a graph representing the wire

harness. Hence **RQ2** was crafted to instigate different network architectures able to extract topological information. The last research question will investigate the transformation of an image of a wire harness into an abstract representation.

- **RQ1**: What kind of visual features can be used to represent a wire harness in an abstract way?
- **RQ2**: What kind of network architectures are suitable for extracting topological information from a wire harness image?
- **RQ3**: How can you classify a wire harness to previously known harnesses?

1.5 Scope and Delimitations

The scope of this study is to develop and evaluate different methods for converting an image of a wire harness into a graph representation and classification of a wire harness. The study will evaluate existing techniques, tweak and combine them to create a pipeline able to extract a graph representation and classifying wire harnesses. In this study wiring harnesses with overlap will not be taken into account. Existing network architectures will be evaluated and tweaked toward the problem at hand, a network built entirely from scratch will not be evaluated.

1.6 AI Disclaimer

ChatGPT was used for text generation to ensure that the text does not contain spelling and grammatical errors. Its also been used for suggesting rephrasing without affecting the core content of this study. All AI-generated content has been critically reviewed by the author taking full responsibility for any content generated by ChatGPT. ChatGPT has also been used in collaboration with the author to solve implementation issues of machine learning models written in Python.

1.7 Thesis outline

Chapter 1: Outlines the background, problem description, purpose, goal, research questions, and the scope and delimitations.

Chapter 2: Introduces some fundamental knowledge of machine learning and key concepts used later in the report.

Chapter 3: Details the overall methodology of the paper.

Chapter 4: Describes the approach and steps in the systematic literature review.

Chapter 5: Presents the findings of the systematic literature review and ends with a comparison listing current gaps in similar studies.

Chapter 6: Explains the experimental study conducted to implement current models and implementing a custom pipeline to extract topological information from a wire harness image and match against known harnesses.

Chapter 7: Showcases the results of the experimental study.

Chapter 8: Discussion of the results of the systematic literature review and the experimental study.

Chapter 9: Conclusion and future work.

2
Theory

2.1 Machine learning

Artificial intelligence is part of the computer science field and has gained significant attention in recent years. Its goal is to mimic the behaviour of the human brain. The set of methods to train a computer to recognize patterns in data is called machine learning (ML). A popular part of machine learning is deep learning, it is an architecture that consists of several hidden layers [26].

Machine learning algorithms

There are four types of machine learning algorithms that most ML algorithms can be divided into: supervised, unsupervised, reinforcement, and semi-supervised machine learning. *Supervised machine learning* includes labeled data, a labeled dataset is made up of input points and a label which represents the ground truth. The ground truth is the desired output from the network given a certain input. One part of supervised learning is classification, given an input, trying to classify what class it belongs to. Segmentation is a type classification, where, given an image, the model tries to predict which pixel in the image is a certain object. This can be in binary form 0 (no) or 1 (yes), or multiclass classification where the model outputs a probability for each class, the sum of all probabilities should sum up to 1. The predicted value of the model can then be compared with ground truth in order to update the model with the knowledge if the predicted answer was correct or not [29]. Compared to supervised learning, *Unsupervised machine learning* consists of unlabeled data. This method is mainly used to cluster data points into different classes. An example is K-means clustering where the algorithm tries to group the input data points into K different classes [29]. *Reinforcement machine learning (RL)* algorithm trains an agent in a given environment, it learns by interacting with the environment though taking different actions and evaluating the result of the action. The purpose of RL training is to maximize the reward function after certain steps of actions [10]. Creating a model capable of playing a game like Tetris would be a suitable task for a RL model. The idea of *Semi-supervised machine learning* is to use both labeled and unlabeled data in order to train a network. Often the unlabeled data makes up the larger part of the data. An example where this is useful is e-mail spam, where a user marks some of the emails as spam but not all [18].

Neural network

A neural network (NN) is a subpart of ML that mimics the neurons and connections of the human brain. Figure 2.1 shows a basic architecture of a fully connected neural network. It consists of an input layer with three neurons that receives the input. A hidden layer with four neurons, where each neuron is connected to all neurons in both the input and output layer. Each connection has its own weight, the value of this weight will determine how much each node affects the output. Eventually, one of the output neurons is activated, indicating the class of the input.

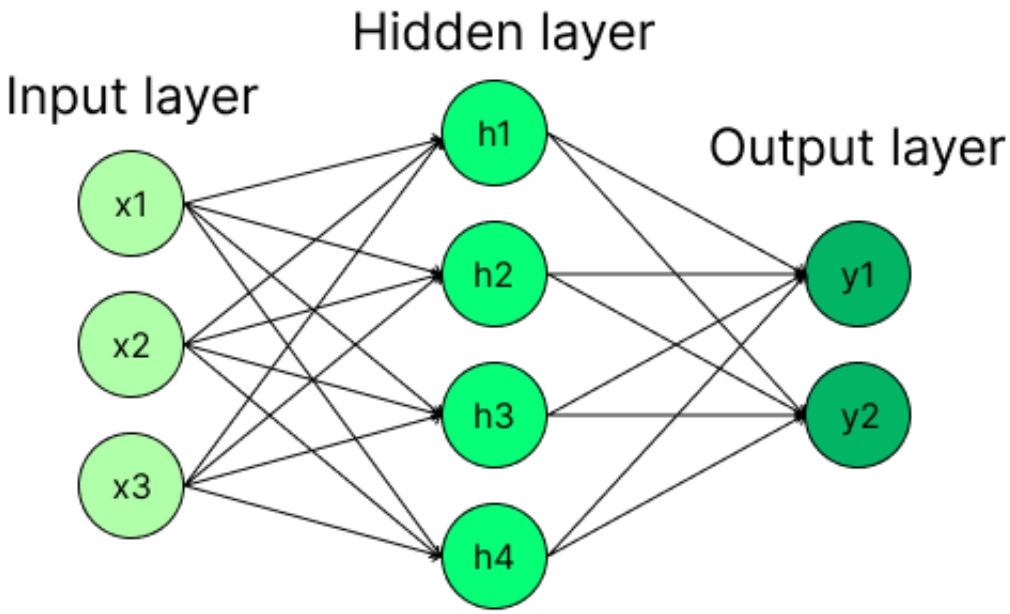


Figure 2.1: Example structure of a neural network

Training

The goal of training a neural network is to minimize the loss functions of the network. The loss functions are differentiable functions that compare the difference between the ground truth label of the input data and the predicted label from the network during a forward pass. An example of a loss function is the mean squared error (MSE).

$$MSE = \frac{1}{n}\sum_{i=i}^{n}(y_i - \tilde{y}_i)\check{\text{s}} \tag{2.1}$$

MSE calculates the mean squared error from all predicted values and true values. When training neural networks, the input gets passed to the network and goes trough all its layers until it reaches the output, this is called a forward pass. When the forward pass is completed and the predicted label is known, the loss is computed. Then with the help of backward propagation, calculates the gradients of each weight and its impact on the loss. Finally the loss regarding each weight is used by an optimizer in order to update the weights to perform better in the next iteration, this completes the training cycle Figure 2.2 illustrates the training cycle with code.

```
model = GraphExtractor()

// Forward pass
prediction = model.forward(image)

// Calculate the loss
loss = model.calculate_loss(prediction, ground_truth)

// Update the models parameters
model.learn(loss)

```

Figure 2.2: Code indicating the key steps for training an machine learning model

Evaluation

The evaluation of a neural network can be done in many different ways. Intersection over Union (IoU) is a metric within computer vision that compares the ground truth bounding box or segmentation of an object to the predicted bounding box or segmentation by the model. The IoU is calculated by dividing the Area of Overlap by Area of Union Figure 2.3. Mean average precision (mAP) is another similar evaluation metric for object detection. It gives some insight into the mean of all classes, where you measure four different values together, called the confusion matrix, IoU, precision in (2.3), and recall in (2.4). The confusion matrix in Figure 2.4 depends on 4 different values:

- True negatives (neither the predicted label or actual is the class)
- False negatives (the predicted is not the class but actual is)
- True positives (both predicted and actual label is the class)
- False positive (the predicted is the class but not the label)

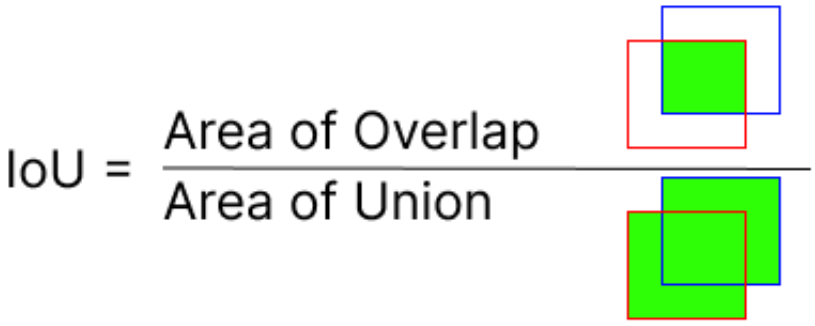


Figure 2.3: Illustration of Intersection over Union equation

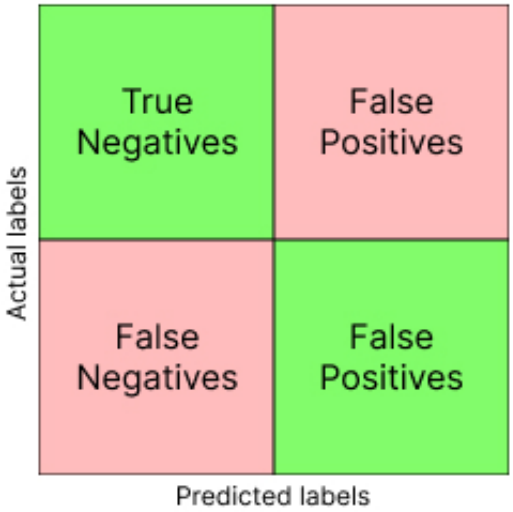


Figure 2.4: Visual representation of the confusion matrix

From the confusion matrix other valuable evaluation metrics like accuracy, precision and recall can be calculated:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{2.2}$$

$$\text{Precision} = \frac{TP}{FP + TP} \tag{2.3}$$

$$\text{Recall} = \frac{TP}{FN + TP} \tag{2.4}$$

There are endless ways to evaluate a model, most popular being the standard ones mentioned above, but also many custom evaluation formulas tailored to a specific task exist. All evaluation methods, however, have the same goal: to get the most accurate and meaningful assessment of how well the models perform on the task at hand.

2.2 Application

There is no machine learning architecture that is best suited for every application. Knowing the expected output of the network is important in deciding the best architecture for the specific problem. For a model that recognizes dogs in images, a CNN would be well suited. However, if the expected output of the model is to remove the dog from an image, a semantic segmentation model would be preferred.

Object detection

There are different types of object detection methods, such as two-stage and one-stage detectors. In one stage both object classification and bounding-box regression are done directly without any pre-generated region proposals. In two-stage detection, the model first identifies regions of interest and then applies the bounding box

and classifies what object it is [21]. One problem that the early object detection algorithms faced was that the region proposal algorithms was a bottleneck for the computation speed.
A convolutional neural network is inspired by the feedforward neural network except for that it utilizes filters to learn different features within the image. Compared to traditional classification algorithms, they require very little pre-processing of the images. CNNs are very good at object recognition by applying convolutions and pooling layers. CNN's ability to recognize patterns is very good. These patterns are recognized with the help of the filter, kernel and stride operations [15]. An example of object detection applications today are self-driving cars, where the car must detect objects surrounding the car, like traffic lights, pedestrians and stop signs [25].

Image segmentation

Image segmentation has had a rapid growth in recent years because of its ability to tell you exactly where the object is. Compared to earlier object detection techniques where a bounding-box is predicted around the predicted object. Image segmentation has the ability to classify specific regions of image into different classes. Especially in cases where more than just the approximate position of the object is necessary. A segmentation of an object identifies exactly what pixels a certain object in the image belongs to that object. This gives the ability to analyse properties of the object better. Such as the color of the object, shape and length. Popular semantic segmentation architectures are U-Net [28] and LinkNet [3] which are all built upon the CNN structure. YOLO is also a very popular image segmentation network. Image segmentation applications are very popular in medical imaging. There are applications that from a CT scan is able to predict whether cancer is benign or malignant [28].

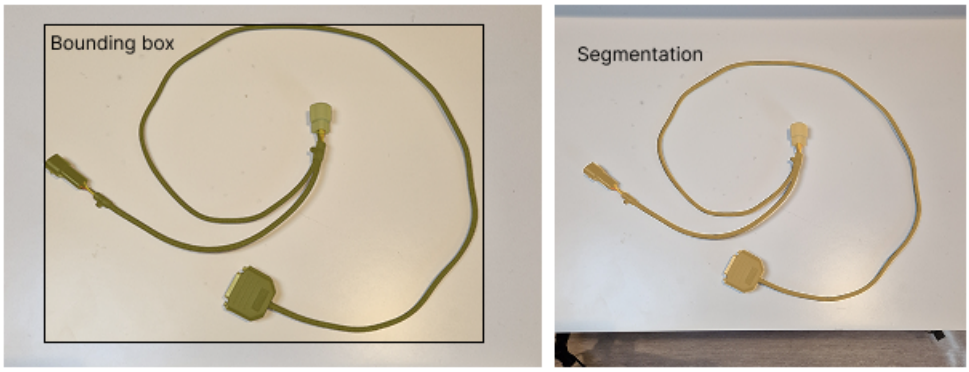


Figure 2.5: Difference between object detection and semantic segmentation on the same harness

Graph classification

An issue with trying to classify a wire harness without and artificial markers using object detection or segmentation is hard because they are very good at learning

local visual features and spatial patterns. However, for classification of a graph they don't understand that a wire is connected to a Connector. To be able to learn and classify relationships like that Graph Neural Networks (GNNs) and Graph Convolutional Networks (GCNs) are two machine learning architectures that are much more suitable. The goal of these architectures is to predict a label for an entire graph, e.g a graph representation of a wire harness. They are especially good for topological reasoning and understand how components are connected to each other [13].

2.3 Machine learning architecture

Deciding on if the model should involve object detection, classification or segmentation, needs to be decided before choosing a network architecture, all network architecture have different strengths and weaknesses. Knowing what the model should output is the first step in answering the previous question, but knowing what is the best method to achieve that result is the second step. In recent years several machine learning models have become widely adopted and used in many applications because of their strengths.

U-Net

U-Net is an architecture initially developed for biomedical image segmentation. U-Net requires pixel-level segmentation and not traditional bounding boxes. The network consists of two phases, an encoder and a decoder. The encoder receives the input and contracts it with max pooling 2x2, in order to capture context. The decoder then expands the input by up-convoluting 2x2 to capture localization. Due to its heavy data augmentation, it requires less data to be trained. U-Net does not require any pre-training on large datasets like "ImageNet", which makes it more suitable to be trained on specific tasks like medical images and wiring harnesses [28].

Feature Pyramid Network

The Feature Pyramid Network (FPN) was developed in 2017 by Facebook AI researchers (Meta). Compared to normal image pyramids that depend on multiple scales of an input image, FPN can build a feature pyramid with the help of a single deep neural network. This helps reduce the amount of computation. The architecture start with a bottom-up pathway; this helps capture both high- and low-level features. Then it up-samples the deeper feature maps to enhance the features in the images. With lateral connections, high-resolution images are merged with low-level feature maps in order to combine semantic information and spatial details. This architecture can detect objects on multiple scales, such as small wires and large connectors [16]. FPN has a similar structure to U-Net, but the predictions differ where U-Net outputs a prediction of a segmentation mask at the highest level and despite a few training images perform well [20].

LinkNet

The application for LinkNet is semantic segmentation. Spatial information is often lost in the encoder because of pooling and strided convolution. U-Net is an example of an architecture where the spatial information is lost [3]. LinkNet is fast and optimal for real-time segmentation [20].

YOLO

Finally YOLO, (You only look once) consists of a single neural network that predicts bounding boxes and class probabilities directly from an image in one evaluation. It divides the image into an $S * S$ grid. Each cell predicts B bounding boxes. Each bounding box consists of 5 predictions (x, y, w, h) and confidence. It also predicts C conditional class probabilities. So the tensor size is $S * S * (B * 5 + C)$. YOLO can very accurately perform single-shot multi-object detection. That is, it can locate multiple objects within a picture in one single stage [27].

2.4 Previous work

Previous work has been conducted within the wire harness community. In [23] they investigate different ways of extracting a topological graph representing a wire harness from an image. They compare *direct* versus *indirect* topology extraction. Indirect topology extraction consists of two steps, starting off by predicting bounding boxes for all wire segments. During the second stage it tries to predict keypoints on the wire segment for each bounding box. Direct topology extraction treats the harness as one whole harness instead of different wire segments. And from the entire harness predict keypoints on the wires. From these keypoints, [23] proposes a method involving spline interpolation to estimate the structure of the wire. Studies focusing on image segmentation have also been tried, [20] compares different image segmentation methods to classify the different parts of a wire harness, in order to predict the harness type. The different methods compared are LinkNet, U-Net and FPN, the best achieving model in their case was FPN with SE-ResNeXt101 as the backbone which achieved an IoU of 86.2%. However, this relies on markers and features other than the visual clues of the harness. The harnesses tested in this paper were from a car door and is mounted on a foam board with clips. The model would most likely perform worse in a different setting where the harness lies on a table and is not mounted in a specific position. [4], [2], [12] does not focus on wire harnesses but DLOs so their work is still relevant. They all try to predict the configuration of wires from a segmentation mask or image of wires. They all consider the inference time, to maximize the FPS in an application. [4] proposes a skeleton traversal strategy to minimize the bending energy to predict the individual wire segments in real-time, even in cases of overlapping wires.

2.5 Graph Theory

There are multiple abstract ways to describe a wire harness, one that is obvious is a graph. Graphs are represented by nodes and edges, e wire in the harness would

be represented as an edge and each connector as a node. There are directed and undirected graphs. Directed graphs have a direction $a \to b \neq b \to a$ while undirected graphs are like two-way streets $a \to b = b \to a$. The direction does not matter for a wire harness, due to the cable being equal in both direction. To encode more information in the graph a weighted graph is useful, which includes a weight for each edge. This would be the length of the wire. Isomorphic graphs are graphs that have equivalent edges. If two graphs have a bijective function $V(G) \to V(H)$ such that any two vertices of G u, v that are adjacent in G iff $f(u), f(v)$ are adjacent in H, as seen in Figure 2.6. To decide if two graphs are equivalent, knowing if they are isomorphic is not enough. For two graphs $G = (V_g, E_g)$ and $H = (V_g, E_g)$, it must follow the conditions: All nodes must have the same label and an edge that exists between two nodes in E must exist in H

$$V_g = H_g \tag{2.5}$$

$$E_g = H_g \tag{2.6}$$

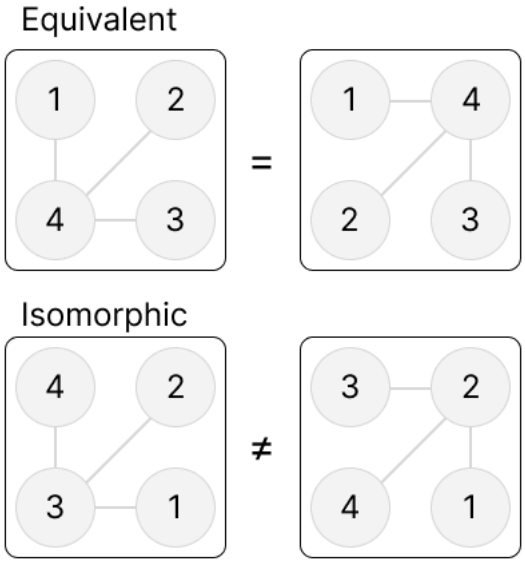


Figure 2.6: Image visualizing the difference between Equivalent and Isomorphic graphs

3
Methodology

3.1 Overview

This chapter describes the research design and methodology for this study. A multiple-method research is performed, starting with a quantitative systematic literature review and a quantitative experimental study. Both follows well established guidelines in order to ensure scientific rigour and reproducibility. The systematic literature review follows Kitchenham's guidelines for a systematic literature review for software development [14]. The experimental study follows the guidelines presented Peffers et al.'s Design Science Research Methodology (DSRM) [24].

3.2 Research design

To answer the research questions in order to fulfill the aim of this paper, a mixed-method research approach was selected. In order to gain knowledge of previous studies strengths and identify weaknesses. A systematic literature review aims to answer RQ1 and partially help to gain some knowledge of network architectures suitable for extracting topological information from an image of a wire harness. Succeeding the SLR, an experimental study will be performed in order to compare different existing solutions and evaluate them against a novel method to extract an abstract representation and answering RQ2 and RQ3.

Research Questions

- **RQ1**: What kind of visual features can be used to represent a wire harness in an abstract way?
- **RQ2**: What kind of network architectures are suitable for extracting topological information from a wire harness image?
- **RQ3**: How can you classify a wire harness to previously known harnesses?

3.3 Research Methods

The following section explains the methods for the SLR and the experimental study.

3.3.1 Systematic literature review

A systematic literature review (SLR) was performed to gain knowledge of previous studies in similar areas. The review helps primarily to answer what kind of features best describe a wire harness and which network architectures are suitable for extracting topological information. The SLR will follow the guidelines of Kitchenham's [14]. To help mitigate common issues such as publication bias, forward and backward snowballing will be performed [8]. Figure 3.1 illustrates the different steps described in [14]:

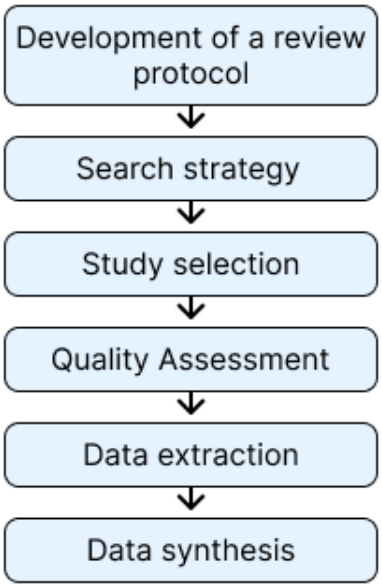


Figure 3.1: Overview of the methodology for the systematic literature review

- **Development of review protocol** Define the scope, search terms, inclusion and exclusion criteria
- **Search strategy** Search strategy to identify primary papers, define database
- **Study Selection** Filter all found studies based on defined inclusion and exclusion criteria
- **Quality Assessment** Evaluate the quality of the paper based on predefined checklist determined in the protocol.
- **Data extraction** Extract data from the studies that have been defined in the protocol, eg. accuracy of model.
- **Data synthesis** Present the data from the primary papers and analyse the findings.

3.3.2 Experimental study

To help answer RQ2 and RQ3, which type of model architecture is best suited to extract and classify a graph representation from an image of a wire harness, an experimental study was conducted. In which existing models were evaluated and tweaked to create an optimal model that performs best on the provided dataset.

The experimental study will follow the standard guideline of Peffers et al.'s Design Science Research Methodology (DSRM) [24]. This research design is especially tailored for solution-oriented research in software development. Starting off with problem identification and object definition, which is already described in the SLR. The next stage of the design phase is where this experimental study will start. Design and development of an artifact. The artifact in this experimental will be a pipeline of machine learning models for graph extraction and classification of a wire harness. After the development stage, demonstration and evaluation will provide informative insights of the artifact and allow further refinements. It consists of iterative development and evaluation of artifacts which is ideal for developing an ML model. The iterative process allows for better quality and efficiency of the final artifact.

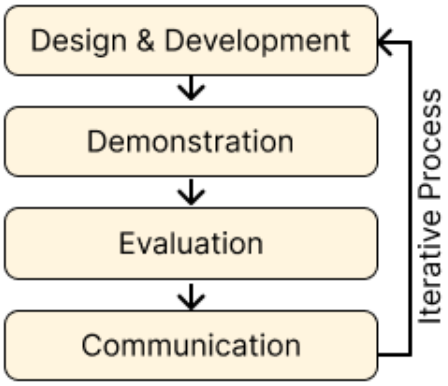


Figure 3.2: The methodology for the experimental study

Figure 3.2 illustrates the four steps outlined in [24].

- **Design & Development**
  The study will involve a pipeline for wire harness representation and classification, guided by the results from the SLR.
- **Demonstration**
  The developed pipeline will be applied to an open source and Wiretronics proprietary dataset to demonstrate its capabilities.
- **Evaluation**
  The artifact will be evaluated with the same evaluation metrics as previous studies in order to compare them against each other.
- **Communication**
  All the results are documented within this study.

3.4 Research quality

To ensure internal validity during the review clear inclusion / exclusion criteria were decided before conducting the review in order to minimize research bias. A screening process of peer-reviewed sources and snowballing to avoid publication bias and ensure that no relevant papers got excluded. Regarding external validity, the

findings from this study can help future work in classifying wire harnesses from an image, quality assurance and robotic assembly of wiring harnesses. The abstract representation of BDLOs is not only applicable to wire harnesses, there are other areas like ropes and textile manufacturing.

To validate the reliability, the study can be reproduced on the Scopus database, which returns consistent result for the same search string. Another research could follow the same steps provided above and come to the same results. Scopus is a big database, but no database captures all, to help mitigate this, snowballing, was used [8]. 42 papers were excluded because they were not in English or Swedish so some relevant papers might have been filtered out during the screening process. Since only one researcher conducted the screening unintentional biases might have occurred.

To ensure research quality for the experimental study clear evaluation criteria are stated. The dataset carefully curated, no overlapping wires within the dataset, all images annotated manually. Using transfer learning from a model trained on a large dataset (COCO [17]) helps to avoid overfitting. The different models developed in this study are tested with the same train/test/val (70/20/10) splits. Internal validity is ensured by verifying different parts of the pipeline by evaluating them independently. By testing on both real data and synthetic data external validity is considered. Reproducibility is ensured by documenting all the steps in this report from data collection to model evaluation.

4
Systematic literature review

4.1 Review protocol

To gain a deeper understanding of the different features of a wire harness and network architectures suitable to extract topological information. The systematic literature review will follow the steps explained in [14], which is a publicly available template for conducting a systematic literature review aimed at software engineering.

- **RQ1**: What kind of visual features can be used to represent a wire harness in an abstract way?
- **RQ2**: What kind of network architectures are suitable for extracting topological information from a wire harness image?

4.1.1 Search strategy

The search was performed on the "Scopus" database, Scopus was selected because its widely recognized as a standard for engineering education research. Unlike databases like Google Scholar that return different results with same search string, Scopus searches are re-creatable, which is key for a reproducible study. A study was conducted that compared different databases. It showed that Scopus has a higher impact ranking than "Web of Science" and covers more journals within the engineering community [5]. It also has more content and citations to Web of Science and greater overlap with modern databases such as Microsoft Academic [19]. To find primary studies, an initial search string segmentation was performed to gain an estimate of the effectiveness of different keywords. The first search being:

```
"(Wire OR cable*) AND (Harness* OR Bundl*)"
```

However, this returned more than 10,000 results; to narrow it down, this search query was combined with keywords to only extract relevant papers to the research objectives. This ensures that the final search string yields a comprehensive yet manageable number of results. The final query that will be used to search the Scopus database is:

```
(Wire OR Wires OR Cable*) AND (Harness* OR Bundl*) AND
    (representation OR topology OR segmentation OR
                    classification)
```

The first part of the search query "(Wire OR wires OR Cable*) AND (Harness* OR Bundl*)", is matched with keywords in the Title, Abstract, and Keywords. The second part "Representation OR Topology OR segmentation OR classification" is matched with all fields of the papers. In order to include more synonyms to "wire harness" the OR operand was used and to include plural phrases an asterisk operator (*) was included. The final search query produced 766 documents on March 4, 2025. Table 4.1 shows the inclusion and exclusion criteria.

Database	Scopus
Keywords	**Article title, Abstract, Keywords**: (Wire OR Wires OR Cable*) AND (Harness* OR Bundl*) **All fields**: (representation OR topology OR segmentation OR classification)
Search fields	First part: Title, abstract, and Keywords, Second part all fields
Inclusion criteria	Papers that focus on abstract or graphs representation, segmentation, topology prediction or classification of a wire harness or DLOs in an image
Exclusion criteria	Poorly structured reports or language, not about wire harnesses or DLOs
Search date	March 4, 2025

Table 4.1: Overview of the review criteria

4.1.2 Study selection

The inclusion criteria for a primary study will be evaluated by one researcher. The screening process will be performed in two stages. The first stage assesses the title and abstract according to the inclusion criteria. The second stage will be a complete read of the through the papers that were selected according to the inclusion criteria in the first stage. In the second stage, papers that do not align with the research scope will be discarded and papers that meet the inclusion criteria and quality criteria will be included as a primary study. The primary studies will be summarized and compared with each other to find any gaps in current research. Because of the overlap between "Wire harness" and "Deformable linear objects", DLOs were added as inclusion criteria, because techniques that are applied to DLOs can be transferred to wire harness applications.

4.1.3 Quality Assessment

The quality of the papers will be evaluated on the basis of different attributes. For the language criteria, the paper must be written in English or Swedish for the reader to understand. The paper must also have some relevance to the research

questions. One reason for choosing the Scopus database is because the papers that are published are peer reviewed. To ensure the quality of the study, it must follow a clear structure and have some form of scientific value. An example of a paper that would be included is an English, peer reviewed document about abstractly matching a wire harness from an image.

- **Language and structure** – Is the text in Swedish or English?
- **Peer-review status** – Has the paper gone though a peer-review process?
- **Relevance** – Is the study relevant to the research questions?
- **Content quality** – Does the text have a clear methodology and scientific rigour?

4.1.4 Publication bias

Since only one researcher is conducting the review and only English or Swedish papers are reviewed and Scopus is the only selected database, there is a risk of publication bias. To help mitigate the publication bias, snowballing method will be performed to help reduce the publication bias [8]. Both forward and backward snowballing will be carried out. The snowballing method consists of identifying additional relevant literature that might have been missed during the initial scoping. Forward snowballing investigates all references cited in the primary studies, and backward snowballing consists of searching for studies that cite any of the primary papers [8]. Even though this helps to reduce the publication bias there is still some bias towards well written Swedish/English papers from top journals.

4.1.5 Data Extraction

From the primary studies, the following information is extracted and presented in a table 5.1: publication year, method used, and key findings. This is extracted manually and compared with each other to present common approaches and gaps in current research and techniques that can be further worked on in this study.

5

Results - Systematic literature review

The following chapter presents the results of the systematic literature review. The data synthesis summarizes all the 12 primary papers from the systematic literature review in Table 5.1. It consists of 5 columns Reference number, Year, Method, Key findings.

5.1 Conducting the review

The first stage of screening included reading the title and abstract of 766 documents. All were evaluated according to inclusion criteria. After filtering the initial studies to show only English or Swedish papers, 42 articles were filtered due to a language barrier. Of the 724 existing articles, the first screening stage began. This involved reading the title and abstract and evaluating according to the inclusion criteria whether or not the articles should go to the second screening. After the first screening 695 papers were discarded because they did not meet the inclusion or quality criteria. 29 articles were taken to the second stage of the screening. However, after completing a full read through, eight papers were selected as primary studies based on the inclusion criteria and study quality. From the primary studies, forward and backward snowballing was performed. This resulted in four more papers being included. Figure 5.1 illustrates a Prisma flow chart of the review screening stages.

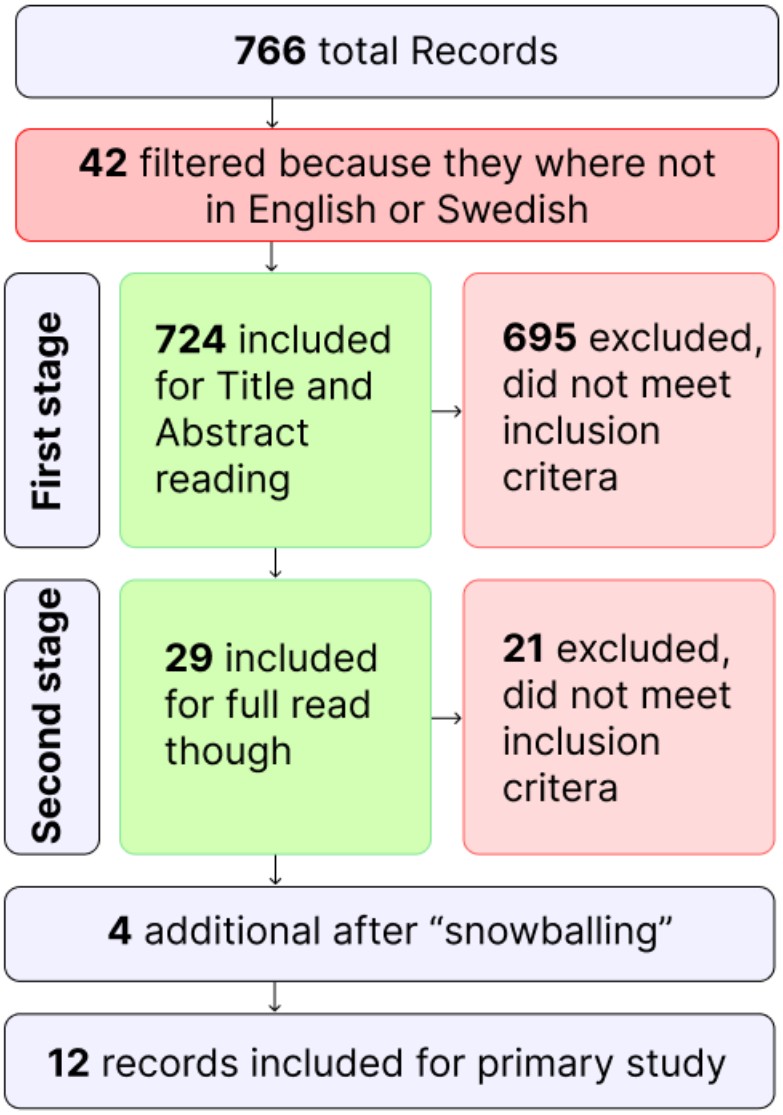


Figure 5.1: Prisma flow chart of the review process

5.2 Data Synthesis

Ref	Year	Method / Model	Key Findings
[23]	2024	Swin-Transformer Panoptic Spline Predictor (SPS), YOLOv8-Pose	SPS model outperforms YOLOv8-Pose in direct topology prediction using keypoint detection and spline interpolation. Achieved mAP@50-95 of 82.5%.
[37]	2023	Graph segmentation and topology matching of BDLOs	Graph-based approach enables abstract comparison of wiring harness topology. Matching performed via topology graphs. achieved an topology matching accuracy of 96.3%
[36]	2023	Instance segmentation (Mask R-CNN, DeepLabV3+)	Extracts topological features (wires and connectors) for downstream robotic manipulation (e.g., gripping).
[1]	2023	DLO3DS: Multi-view 2D camera, B-spline modeling + stereo triangulation	Accurate 3D reconstruction of DLOs from 2D views using robotic-arm-mounted camera.
[33]	2024	Particle Filter with B-spline tracking on depth images	Per-branch tracking without segmentation enables robust BDLO tracking under occlusions and low contrast.
[7]	2024	OpenCV + Agglomerative Clustering	Uses clustering and vision to detect grip points along harness branches for robot interaction.
[22]	2021	CNN + depth profile correction for 3D path extraction	Enables robot-guided harness assembly using 3D profiling from grayscale and depth input. Demonstrated 72.5% success rate in automated assembly tests.
[20]	2021	Deep learning-based semantic segmentation for AOI	Enables in-line inspection of rigid/deformable harness components, improving traceability and QA.
[12]	2023	Skeletonization + B-spline fitting (DLOFTBs)	Fast, accurate shape tracking without deep learning. Handles occlusion, self-intersections.
[11]	2021	CNN with saliency maps for wiring harness classification	Provides interpretable classification of wire branches for robotics guidance.
[2]	2022	Skeleton-based segmentation with similarity learning (FASTDLO)	Fast and efficient segmentation of DLOs using skeleton-based structure, useful for matching harness shapes.
[4]	2023	Minimal Bending Energy Skeleton Traversal (mBEST)	Optimized for real-time, skeleton-based DLO detection by minimizing bending energy. Effective for complex intersections.

Table 5.1: Summary of Primary Studies and Key Findings

The following tables list the primary papers divided into four different tables. All focus on some part of the wire harness or DLO classification or segmentation, topology prediction or matching, and feature extraction. All tables consist of the same information: paper, model/method used, task, metric, and metric score from the results of the primary studies. Table 5.2 compares studies that focus on matching a known topology to an image of a wire harness and topological prediction of a wire harness. Table 5.3 lists the results between the studies that focus on image segmentation and instance segmentation of electrical wires and wire harnesses. The following table, Table 5.4 shows the quantitative results of the studies that focus on feature extraction. Two of the features that both papers have in common are endpoints and branch points (junctions). The last table, Table 5.5 shows the results from [11]. This study tries to classify a wire harness from an image.

Paper	Model/Method	Task	Metric	Value
[37]	Graph-based matching	Topology matching	Topology accuracy	96.3%
[23]	YOLOv8-Pose	Topology prediction	mAP@50-95	89.8%

Table 5.2: Comparison of primary papers focusing on matching and prediction of wire harness topologies

Paper	Model/Method	Task	Metric	Value
[4]	mBEST	DLO detection and segmentation	DICE	92.17%
[2]	U-Net variant	Instance segmentation	mIoU	77.0%
[36]	Not specified	Image segmentation	mAP@50	89%

Table 5.3: Results from the image and instance segmentation papers

Paper	Model/Method	Task	Metric	Value
[33]	Particle filter with B-spline	Endpoint/junction prediction	mAP@50-95	88.5%
[20]	Mask R-CNN	Feature segmentation	Accuracy	85.55%
[22]	Masked R-CNN	Feature extraction	Accuracy	94%

Table 5.4: Results of the primary papers focusing on feature extraction

Paper	Model/Method	Task	Metric	Value
[11]	CNN variations	Classification	Accuracy	96.76%

Table 5.5: Table of the primary papers that focus on classification of a wire harness

5.3 Graph representation

To predict the topology of a deformable linear object there are two main strategies, direct and indirect topology prediction. Direct topology prediction can without a region of interest step predict keypoints on a BDLO. Meanwhile, indirect topology prediction requires a region of interest such as the prediction of bounding boxes around the wire segments. After bounding box prediction of all wire segment the model tries to predict keypoints on the wires.

A deep learning method for direct topology prediction is introduced by [23], they propose a novel Swin-transformer model (SPS). The model predicts keypoints on each segment from an image of a wire harness, then each keypoint is interpolated using cubic splines to form a structured representation of the harness. This can be interpreted as a graph with predicted keypoints as nodes and splines as edges. This method managed to outperform a YOLOv8m-pose model in direct topology prediction with a mAP@50-95 of 82.5 % compared to YOLO's 11.8% mAP@50-95. Although this approach is fast and compact, the paper also evaluates indirect topology prediction using YOLOv8m-pose, where a bounding box is predicted around each wire segment, and then keypoints are predicted. This takes more computer power than direct topology prediction, but results in better accuracy, the YOLOv8m-pose model achieved mAP@50-95 of 89.8% contra [23]s 81.1% [23].

Meanwhile, [37] proposes a graph-based matching algorithm that aligns a previously known wire harness topology with a detected topology from a stereo camera image. The known topology is explicitly represented as a directed graph:

$$F = (N^{k+1}, \varepsilon^k, A^k) \tag{5.1}$$

Where N is a set of nodes, ε is a set of segments, k is the number of segments, and A is a set of features. An example of different features are the length and color of the wire segments. Their method uses a segmentation and skeletonization technique, with a minimum spanning tree (MST) method to locate leaf nodes and branch nodes. They achieved 96.3% accuracy, on a dataset limited to no overlapping wires [37].

There are also different methods that do not explicitly state the structure as a graph, FASTDLO [2], mBEST [4], and DLOFTBs [12] output a representation that can be converted to a graph representation even though they are not explicitly described as such. These methods extract a skeleton from a segmentation mask, which inherently defines a set of connected keypoints that can be treated as a graph structure. Future work could explore unifying [37] mathematical graph based abstraction with the segmentation pipelines, enabling better extraction and classification of the DLOs.

5.4 Semantic and instance segmentation

Some of the primary papers focus on segmenting deformable linear objects. [2], [12], [4] are papers that manage to extract individual wires from a BDLO segmentation mask. All three methods rely on a segmentation mask of a DLO, they differ in the algorithms used to estimate the structure from the mask. FASTDLO and mBEST use a CNN to estimate a segmentation mask from the input image, while DLOFTBs takes an already segmented wire mask as input. A skeletonization algorithm is then applied to this mask; this gives the centerline which maintains the connectivity of the DLO.

The main difference between the three methods is how they interpret intersections and reconstruct the structure. FASTDLO uses a neural network to decide how segments should be connected, based on different features: color, thickness, and directional consistency to resolve overlap. DLOFTBs, takes a geometry-driven approach, linking segments based on distance and orientation heuristics. This makes DLOFTBs computationally efficient, as it avoids the need for additional neural network processing. Meanwhile, mBEST fits a spline to a wire skeleton, then traverses the skeleton to find keypoints such as junctions and endpoints. To resolve overlapping wires a minimize bending energy method is used to help ensure correct topology. The minimum bending energy helps with the path generation in order to ensure that the centerline follows the most probable physical structure [4]. This achieves a fast computational time but can only handle one intersection per DLO and relies on a manually tuned threshold based on the number of objects in the image.

Another key difference between these methods is how they represent the final structure of the DLOs. FASTDLO outputs a set of keypoints along each segmented DLO, which can be used for further processing or potential spline fitting. DLOFTBs instead fits a B-spline curve that is continuos which is useful for real-time applications. mBEST, on the other hand, reconstructs the centreline of each DLO by following a minimal bending energy path, which ensures that the structure adheres to physically probable configurations. FASTDLO is capable of segmenting DLOs that is different from electrical wires, as shown in Figure 5.2, where it manages to segment medical wires.

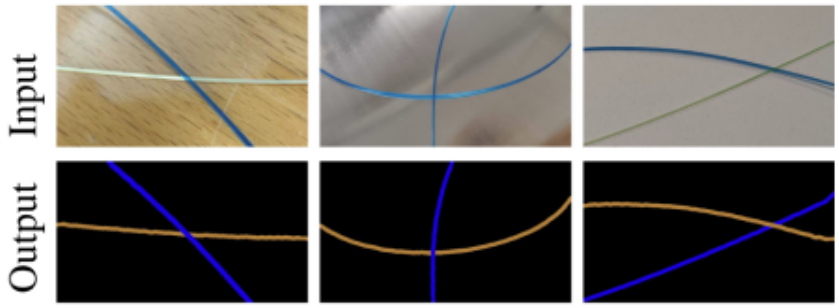


Figure 5.2: Medical wire segmentation from FASTDLO[2]

They all have different evaluation metrics, so comparing exact results is hard. FASTDLO evaluates its segmentation performance using the IoU metric, on their own custom dataset consisting of 135 labeled images, this was evaluated with two different backbone networks, ResNet-50 and ResNet101. ResNet101 provides the best IoU with a result of 77.77 %. This increased to 84.49% when pre-trained on synthetic dataset with chroma-keys. DLOFTBs does not report IoU but instead evaluates the tracking accuracy of its B-spline representation, measuring how well the fitted curve approximates the ground truth shape. mBEST, on the other hand, uses the DICE score, which is similar to IoU but better suited for thin, elongated objects. It also evaluates runtime performance in frames per second (FPS), demonstrating a 50% speed improvement over FASTDLO while maintaining higher segmentation accuracy.

FASTDLO is best for instance segmentation in cluttered environments where multiple DLOs need to be identified and separated. DLOFTBs has very fast computation which fits for real-time applications, mBEST outperforms the others in handling complex DLOs curvatures and overlapping structures. All these methods have different advantages, they all contribute to advancing DLO segmentation and reconstruction, opening new possibilities for robotic manipulation, automation, and industrial applications. They also solve some problems that [33] mention with the large background noise that affects keypoint prediction when dealing with bounding boxes around wire segments. Another study that focused on segmentation is [20], this study evaluated three different architectures and compared their Intersection over Union, this was chosen because it heavily penalizing bad predictions. The architectures compared were U-Net [28], LinkNet [3] and Feature Pyramid Network (FPN) [16]. These models were chosen because of their strength in semantic segmentation and the ability to extract low-level features. Just as [33] mention labeling on a pixel level and using semantic segmentation is very good in for applications regarding deformable linear objects. All models they test use transfer learning with different pre-trained networks, EfficientNet-B3 an ResNet50. The same hyperparameters were used to ensure fair comparison between the networks. The best overall architecture was the FPN with Se-ResNeXt101 as the backbone, averaging around 86.2 % IoU and a precision of 100%. However, this solution trained and tested on one wire harness from a car door. The model predicts the nine different components that the harness consists of: fully taped wires, spiral taped wires, clips, and connectors. The solution is limited to a single wire harness that lies flat in a assembly position with artificial markers as can be seen in Figure 5.3.

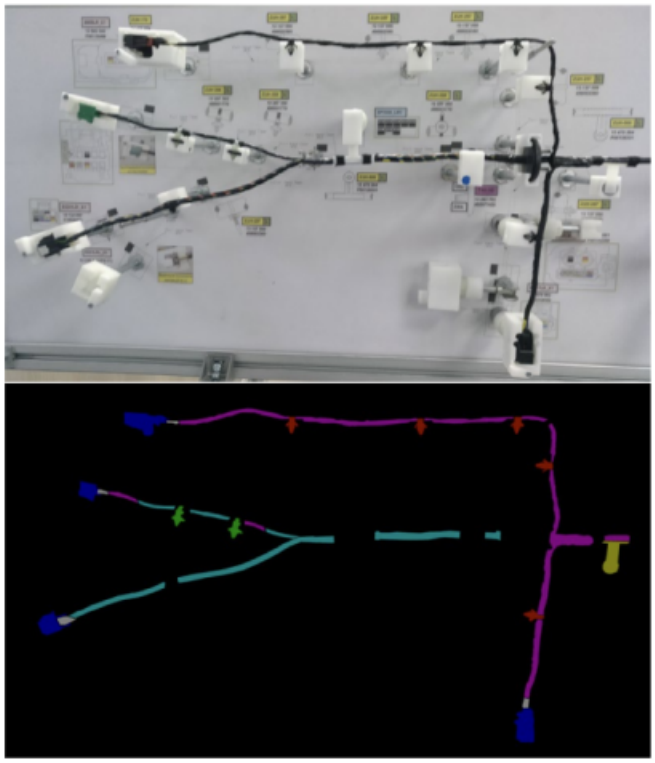


Figure 5.3: Segmentation results of the Volkswagen car door from [20]

5.5 Feature extraction

Earlier DLO tracking methods, that do not depend on segmentation have certain limitations such as having to retrain the network for new wire harnesses or DLOs [33]. [33] method focus on keypoint detection using a YOLOv7-based detector trained on synthetic depth images. Comparable to [23], this solution predicts the topology indirectly. This allows each wire segment of the BDLO to be individually tracked using particle filters. The benefit of this approach is improved generalization to new wire harness topologies, and it also reduces the need of color information of the wires which is common in wire harnesses where all wires are black. However, the keypoints are sometimes inaccurate, believed to be due to the gap between the bounding box around the wire. They suggested that future work should rely on pixel-level annotation method could improve performance.

Another method for obtaining features from a DLO proposed by [22] is a method that combines convolutional neural networks with depth profile correction to extract the 3D path of real-world wiring harnesses. It uses grayscale depth images to guide robotic assembly, achieving a mean absolute error of 8.15 mm. This method is validated in a real industrial context and demonstrates a 72.5% success rate in automated harness assembly tasks. This method may be more precise 3D path extraction than [33] but less generalizable towards new topologies. Another paper that focuses on feature extraction is [1], they introduce a system for reconstructing DLOs by fusing multiple 2D views from a stereo camera mounted on a robotic arm. B-spline

curve method is used on the segmented 2D projections and triangulates their positions to obtain an accurate 3D model, achieving a reconstruction error of 0.82 mm.

Finally, [36] uses a deep learning method using Mask R-CNN, for instance segmentation, to extract important features such as junctions and endpoints in order for robotic manipulation of the BDLO. The Mask R-CNN used ResNet50-FPN backbone pre-trained on the ImageNet dataset. Individual connectors are located with object detection along with shape matching to a CAD model. They used a semi-manual annotation that they mentioned could reduce training time, and by doing this they achieved a reduced training time per image by 248%. They also propose future work using predicted labels extracted by the model and fuse them with depth information to perform segment machining of the BDLO. Figure 5.4 illustrates an example of feature extractions of a wire harness from [36].

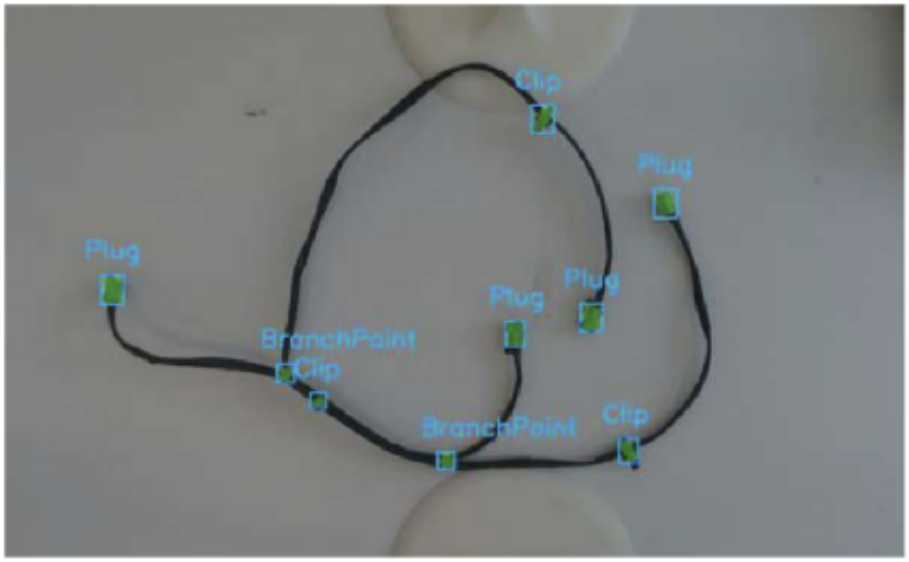


Figure 5.4: Feature extraction of a wire harness from [36]

Figure

5.6 Classification of wire harness

One of the primary studies only focuses on classification is [11], they try to classify four different branches of a wire harness from a Volkswagen Caddy cockpit. They present a deep neural network with saliency maps to classify the different branches, improving the interpretability for downstream robotic tasks. With the help of visual cues, such as shape and color, to distinguish segments. The best performing model achieved an accuracy of 96.67%. A similar paper is [7], but in contrast to [11], focuses on locating grip points along the harness. The method uses OpenCV and agglomerative clustering to find junctions and endpoints where a robot can interact with the harness safely and effectively.

These papers both tries to classify the harness but focus mainly on different things. [11] classifies the harness segment, while [7] identifies where to interact with it.

5.7 Summarization

One thing most of these papers highlight is the lack of a large enough dataset (data scarcity). An comprehensive annotated wire harness dataset would allow for better benchmarking in different studies. Many of the papers also rely on some form of B-spline and spline interpolation to estimate the structure of the harness or DLOs. Estimating the wire structure with interpolation instead of segmenting the entire wire decreases the inference time and is more optimized for real-time application, but sacrifices some accuracy in wire structure.

6
Experimental study

Since there is no way of knowing how well a model will perform on a given dataset without implementing a model and testing it on the dataset an experimental study is performed to measure how well the different models perform. This chapter will introduce the Design & Development of the study, chapter 7 will cover the demonstration and chapter 9 the evaluation. The list below demonstrates the different steps from the research design [24]:

- **Design & Development**
  Two different YOLO-based models will be developed, one involving keypoint detection to extract a graph representation and one segmentation model followed by skeletonization to extract the topology. Following the topology extraction, validate it against known harnesses to classify which harness it is.
- **Demonstration**
  The model is trained and tested on an open source dataset (D2 Figure 6.2) and Wiretronics proprietary dataset (D1 Figure 6.2).
- **Evaluation**
  To evaluate the solution different parts of the pipeline will be evaluated against similar studies regarding that area. For classification of the wire harness accuracy will be measured. Meanwhile, the segmentation part will be evaluated using mAP and IoU to compare the segmentation against previous studies.
- **Communication**
  All the results are documented in this study.

The final objective is to propose a pipeline capable of accurately extracting a graph representation of wire harnesses and being able to match them to an image of a wire harness under realistic conditions.

6.1 Data scarcity

There is data scarcity within the wiring harness community [23], [30]. [38] explains a way to increase the size of a dataset by using the "Copy-Paste" method. In their case they created a simple dataset of wire harness bags and then extracted the harness bag from the image in order to paste onto various backgrounds to simulate the real world with different lightning and shadows. Evaluating the same model trained on only real images versus trained on both real images and synthetic images. The model trained on the synthetic dataset increased the dice score from 0.464 to 0.525 and IoU from 0.374 to 0.411. Another way to mitigate data scarcity is to use transfer

learning. That is, using a model that has been trained on a larger dataset and then training on a smaller dataset to achieve higher accuracies. However, [36] did a study in which they should segment a deformable linear object and decided to use transfer learning of a ResNet50 pre-trained on ImageNet [6]. But in their case it did not perform better because the difference in the datasets, ImageNet lacked some features that deformable linear objects introduce. This required annotating more data to get the model to efficiently capture harnesses. [34] released an open source annotated dataset of 28 646 electrical wires.

6.2 Data collection

Wiretronic provided ten different wire harnesses with a simpler design; each harness consists of three connectors connected by two wires. To generate a dataset, the harnesses were photographed with a Samsung Galaxy S22. This resulted in 82 different images of the wire harnesses. Between each image, the harness was moved around to change its structure. All harnesses where photographed on a flat surface, without any overlapping wires. Another open source dataset of around 1100 images was reduced to 96 images when excluding all images with overlapping wires [23]. This dataset included two different types of wire harness, one with three connectors connected by two wires and one with seven connectors and nine wire segments. The images were then annotated with Wiretronics internal annotation tool. In order to evaluate the different models, two datasets were annotated. One including all images along with polygon annotations, and one dataset with keypoint annotations. The keypoint annotation is done by placing a certain number of keypoints on each wire segment, [23] chose five keypoints per wire segment. In order to train their model on Wiretronics wires, five keypoints per wire segment were chosen to annotate Wiretronics harnesses as well. The polygon annotation involved placing polygons around the entire harness and each pixel group was assigned a label "Wire", "A" etc. as can be seen in Figure 6.1. The annotations were then exported in the YOLO segmentation format. Lastly another open source dataset was downloaded from Kaggle containing 30 000 images of electrical wires provided by [35].

Figure 6.1: Labels for each connector

6.3 Pre processing dataset

All datasets are split up into a 70/20/10 split, 70% training data, 20% validation and 10% test data. All images were resized to 640x640 px, to increase the size of the dataset synthetic images were included. This to help avoid the problem of data scarcity all images are rotated 45 degrees all the way up to 315 degrees. The training data was further augmented using the copy-paste method [38] onto 4 different backgrounds. This helped the model generalize and recognize the object within the images [38].

Datasets

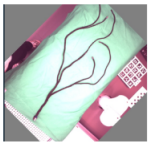
Dataset	Example image	With rotation	With background
D1	82 (images) 	656 (images) 	2624 (images) 
D2	96 (images) 	768 (images) 	3072 (images) 
Electric wires	30 000 (images) 	Already augmented	Already augmented

Figure 6.2: The different datasets used for training models

6.4 Design & Development

6.4.1 Keypoint detection

After the datasets had been generated, the first approach towards predicting the topology of a wire harness was a keypoint approach. It started with downloading [23] publicly available models in order to evaluate them on D1. From scratch this model performed badly and overestimated the amount of wires. To improve the model it was trained further on D1 dataset. With transfer learning from the best performing model from [23] YOLO_D2&D3_A2_PretrainedD4.pt. After that a YOLO pose estimation model was trained from scratch on a combined dataset "D1&D2". Yolo-pose requires a number of keypoints it should estimate per wire segment. To ensure that the model could be trained on D1, D1 was annotated with the same number of keypoints per segment.

6.4.2 Semantic segmentation

As mentioned in 2.5 there are multiple different ways to represent a graph and the one definition that will be used in the study is an undirected weighted graph. This is the one that represents the wire harness the most. Hence the wires doesn't have a direction and the weights for each edges will represent the length of that wire. This is required if two harnesses have the same components and topology but different wire lengths.

The second network implemented in this experimental study consists of a two-stage model. The first stage is a YOLO-11 based segmentation model that extracts the different parts of the harness. These results are then processed with a skeletonization technique to extract the topology, the information is then fed into a Graph Convolutional Network (GCN) which is very good at recognizing graphs. The prediction is then validated against a validation file, this file contains the correct structure for each harness Figure 6.3. The accuracy of the GCN was evaluated with accuracy where it can be correct or incorrect (1 or 0). With the help of the validation file however, even if the GCN isn't accurate again it will return the most plausible harness along with an error list. The error list contains information about missing nodes or edges and wrongly labeled nodes in order to be one of the known harnesses.

Figure 6.3: Architecture of final model

The topology of a harness will always be the same no matter how you place it. To obtain the topology from the segmentation, a skeletonization algorithm similar to mBEST [4] was implemented. Instead of extracting one continuous wire mask from an image, mBEST tries to differentiate between different wires in an image. The proposed solution uses a skeletonization algorithm to narrow the wires down to 1px in width. Followed by a using minimum bending energy to differentiate between different wires. In this study a skeletonization technique was implemented with some differences. It starts off by narrowing down the width of the wire mask to 1-pixel-wide. Then removing shorter branches that can be caused by the segmentation. Following the removal of shorter branches the skeleton is traversed to ensure connectivity, that its one solid skeletonization structure. This is advantageous in non-overlapping wires. From this skeleton traverses the skeleton and assigns endpoints to each part of the skeleton that only includes one neighbour pixel, and classifies each pixel that has three neighbours a junction. This is done together with the input of all the connector positions within the harness. Each endpoint is assigned the closest connector position. Following the skeletonization the generated graph with the help of python package "networkx", all the lengths between the junctions and endpoints are calculated in pixels. To gain an estimate of the lengths of the wire segments. The information from the connector labels, edges and wire segment lengths are then saved in a .json file A.1.
In the initial iteration the topology file was structured so each endpoint was connected to a connector. The connector was presented with the label of the connector like "A". But after a while when training on harnesses with multiple connectors with the same name, it was impossible to know what connector each endpoint was connected to. As visualized in Figure 7.10 in the image below. The figure demonstrates a connector with 4 connectors connected with 5 wire segments. In the initial extraction of the graph representation the label of the connectors was used as endpoints.

This introduced an issue were multiple different topologies could be extracted from that presentation which made it difficult to classify against known harnesses. To solve this problem, the extracted graph was represented with endpoints instead of the label as key. This ensures that the extracted graph representation can only represent one harness. To classify a wire harness with a GCN along with a validation file has not been tested by any previous studies.

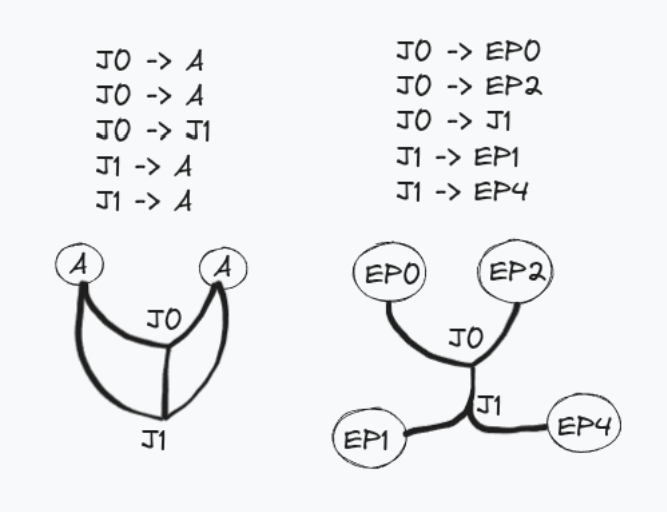


Figure 6.4: Example showing the need for endpoints and not connector id

6.5 Evaluation

The topology model [23] is evaluated with mAP and mAP@50-95 in order to compare it against their results. The final model developed in this study will be evaluated in multiple steps. The segmentation model will be evaluated with IoU, Dice, mAP@50 and precision, to compare it to [20], [2], [4]. The harness prediction GCN model will only be evaluated using precision / accuracy to compare it with [23].

6.6 Experiment setup

Wiretronic provided all compute for the training of the machine learning model. The computer has two "GeForce RTX 3090 24GB" graphics cards, which allowed for at least two trainings to run simultaneously.

7

Results - Experimental study

This chapter presents the results of they experimental study. Its divided into four sections, the first one shows the results from the pose estimation models that predict keypoints along the wire segments 7.1. Following this section, the results of the segmentation models will be presented 7.2, after the segmentation results, the next step in the pipeline will be presented, the skeletonization 7.3. Finally, the GCN model that classifies the extracted graph representation will be demonstrated 7.4.

7.1 Keypoint prediction

This section presents the results of the keypoint prediction models. Table 7.1 presents the evaluation results for each model: model, which dataset the model was tested on, mAP and mAP@50-95.

- **Model 1**: YOLO_D2&D3_A2_PretrainedD4 from [23], without any further training.
- **Model 2**: YOLO11n-pose that uses YOLO_D2&D3_A2_PretrainedD4 as backbone trained on D1.
- **Model 3**: YOLO11n-pose trained on the combined dataset D1&D2.

Model	**Dataset**	**mAP**	**mAP@50-95**
1	D1	0.021	0.002
1	D2	**0.901**	**0.731**
2	D1	0.390	0.210
3	D1&D2	0.610	0.422

Table 7.1: Results of the pose estimation models

Listed below are four different images, from the keypoint prediction models. Figure 7.2 and Figure 7.1 are keypoint predictions by model 1. Figure 7.3 shows the predictions by model 2 and the final Figure 7.4 is model 3's prediction of a wire harness from D1.

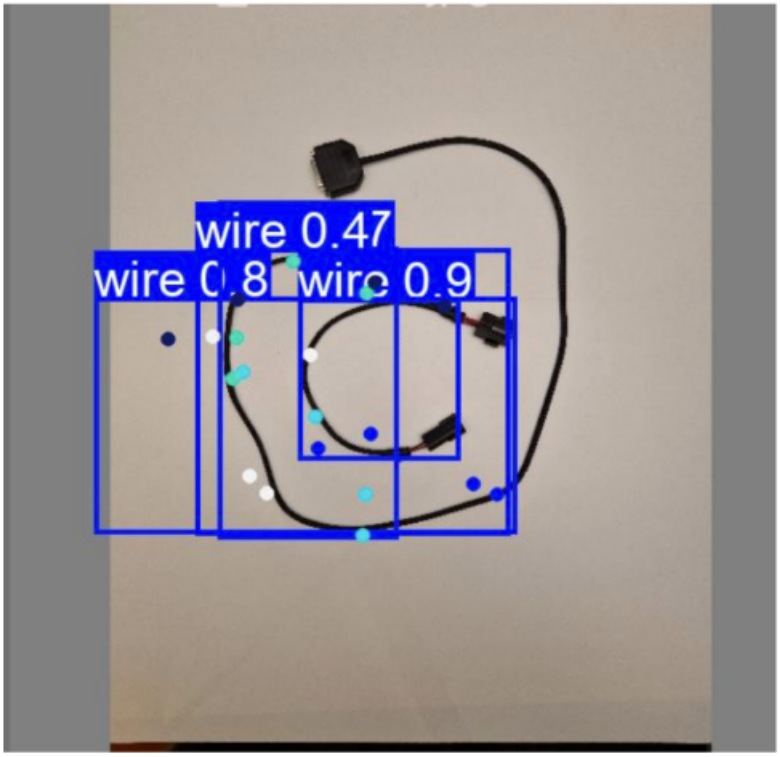


Figure 7.1: Model 1 prediction on an image from D1

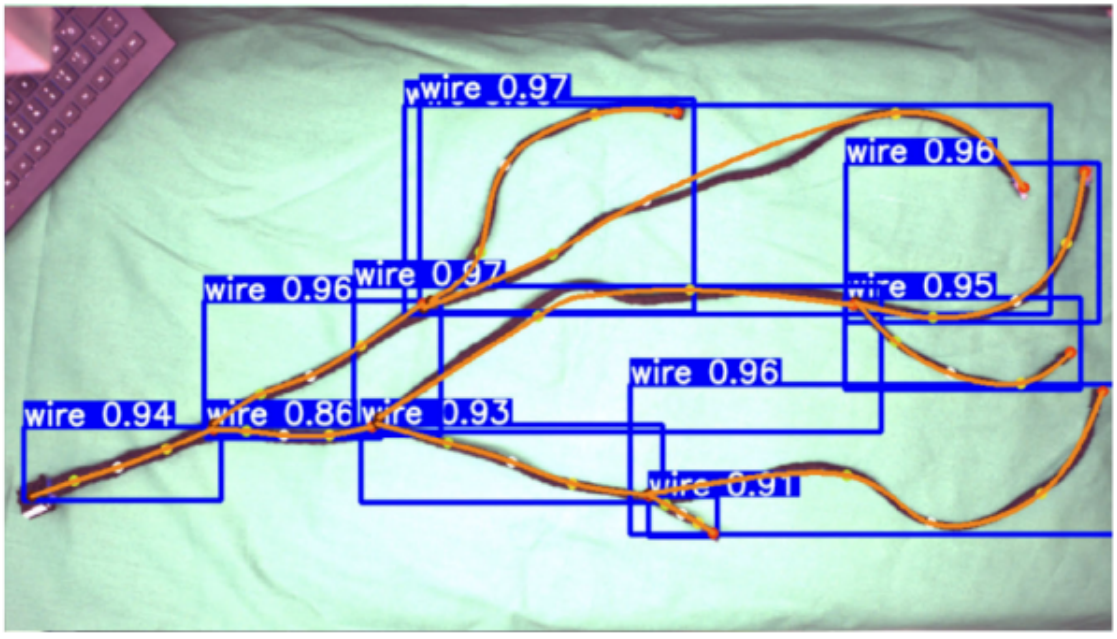


Figure 7.2: Model 1 prediction on an image from D2 with spline interpoolation

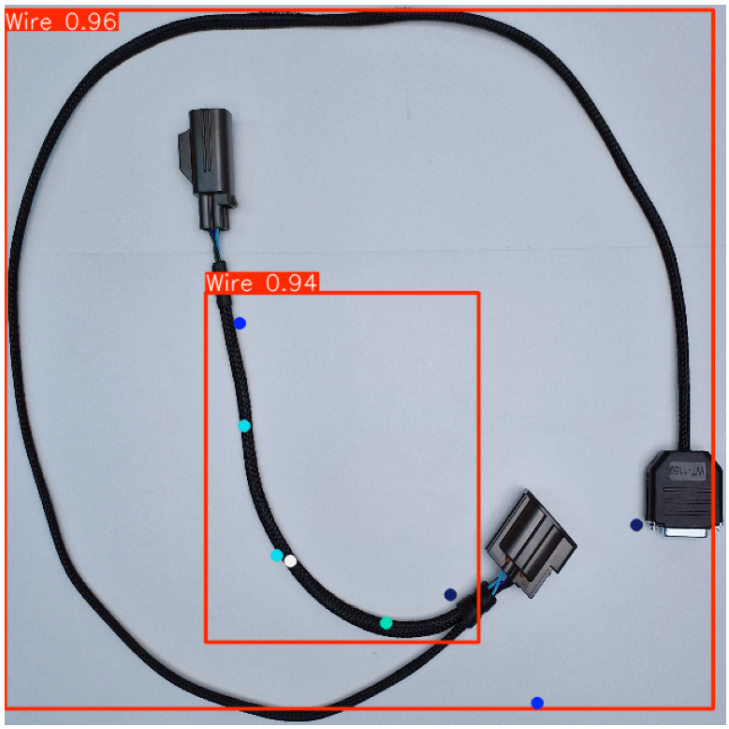


Figure 7.3: Model 2 predictions on an image from D1

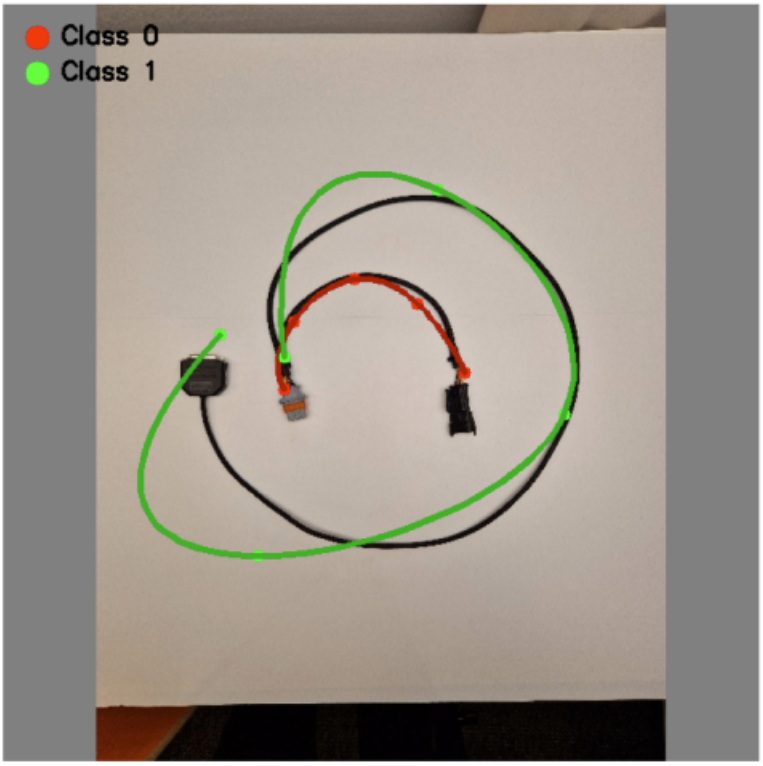


Figure 7.4: Model 3 predictions on an image from D1 with spline interpoolation

7.2 Semantic segmentation

Following the pose estimation models, the segmentation pipeline was implemented. Table 7.2 explains how the different segmentation models were trained and what backbone models they have. The results are split up into three different tables that list: Model, what dataset the model was trained on, precision, recall, F1 Score, mAP@50, mAP@50-95. Table 7.3 shows the results for segmentation on the electrical wires dataset, Table 7.4 shows the results on the combined dataset D1&D2. Finally Table 7.4 shows the evaluation metrics of the different models for the class: Wire.

Model	**Explanation**
1	Yolo11n-seg trained on electrical wires
2	Yolo11n-seg pre-trained on D1&D2
3	Yolo11n-seg trained on electric wires then D1&D2
4	Yolo11n-seg trained electric wires then D1&D2 with initial 10 layers frozen

Table 7.2: Explanation for how each model was trained

Model	**Dataset**	**Precision**	**Recall**	**F1 Score**	**mAP@50**	**mAP@50-95**
1	Electrical wires	**0.996**	**0.945**	**0.970**	**0.967**	**0.910**

Table 7.3: Segmentation results on the electrical wire dataset

Model	**Dataset**	**Precision**	**Recall**	**F1 Score**	**mAP@50**	**mAP@50-95**
2	D1&D2	0.827	0.942	0.881	0.925	0.721
3	D1&D2	**0.837**	**0.938**	**0.885**	**0.973**	**0.734**
4	D1&D2	0.828	0.858	0.843	0.883	0.657

Table 7.4: Segmentation results from models

Model	**Dataset**	**DICE**	**IoU**	**Precision**
1	D1&D2	0.007	0.003	0.004
2	D1&D2	0.962	0.962	0.962
3	D1&D2	0.950	0.904	0.904
4	D1&D2	**0.989**	**0.978**	**0.978**

Table 7.5: Segmentation results from models (only wire)

The images below visualize segmentation results. Figures 7.5, 7.8 and 7.8 are all test images from D1 and D2. The last two figures, Figure 7.6 and Figure 7.9 are

of a custom-designed harness in Figma. It is based on an image from D1 with an additional wire and connector added.

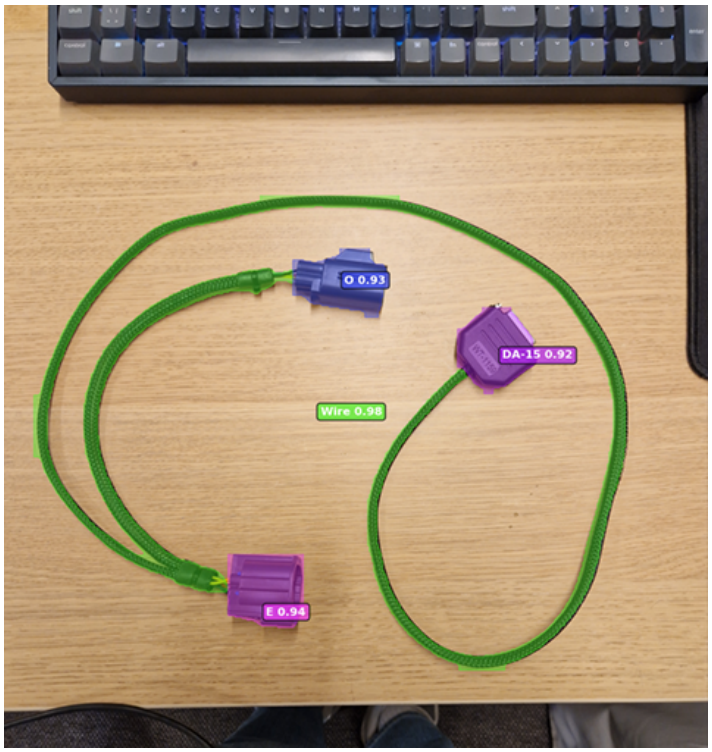


Figure 7.5: Visualization of Model 3 segmentation mask on test image from D1

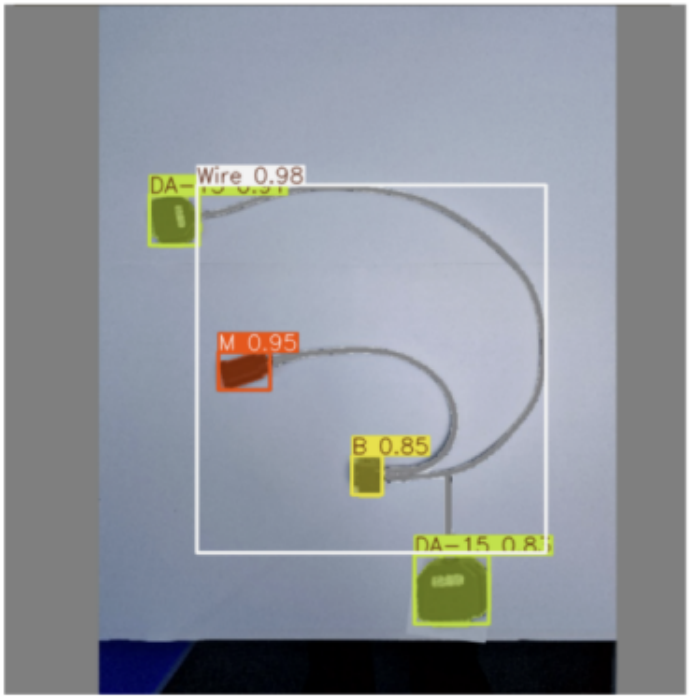


Figure 7.6: Results from Model 3 on an artificial harness

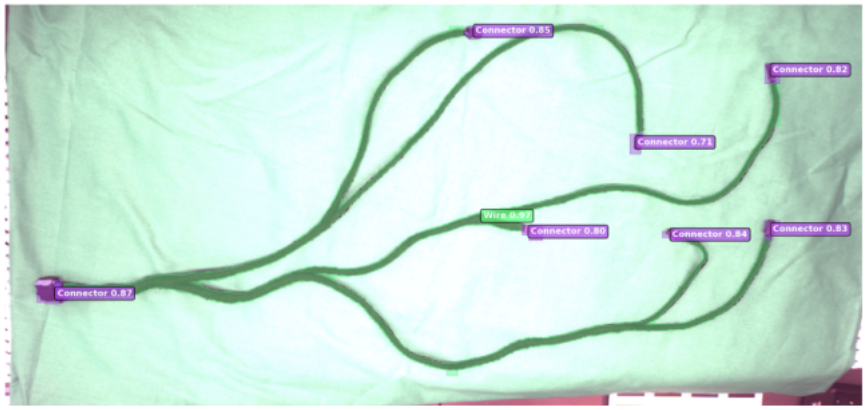


Figure 7.7: Segmentation mask of Model 2 on an image from D2

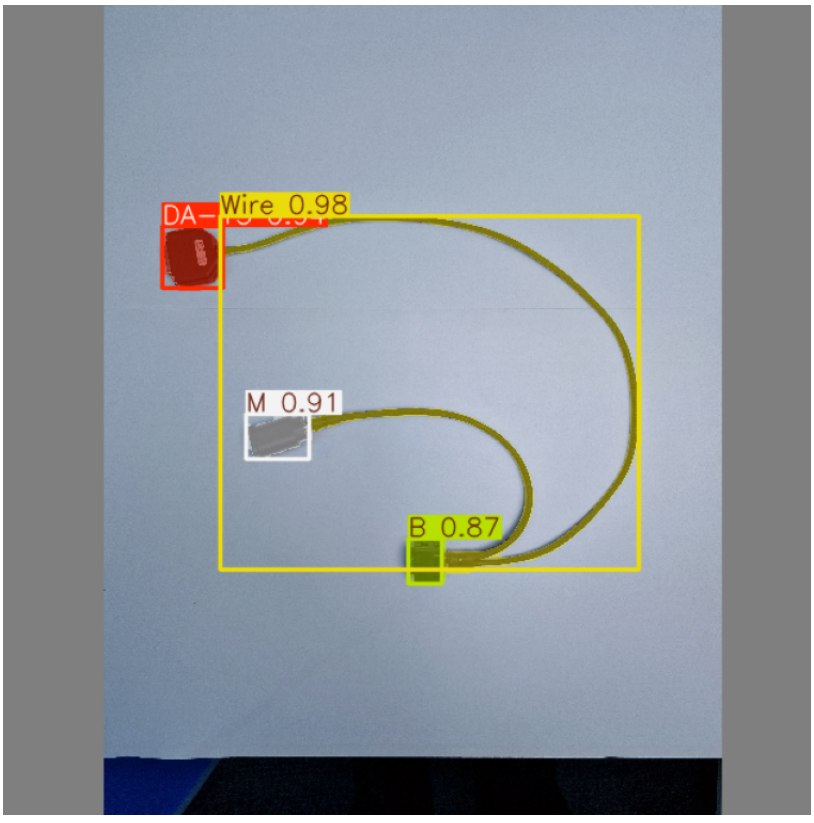


Figure 7.8: Model 2 predicted segmentation mask on an image from D1

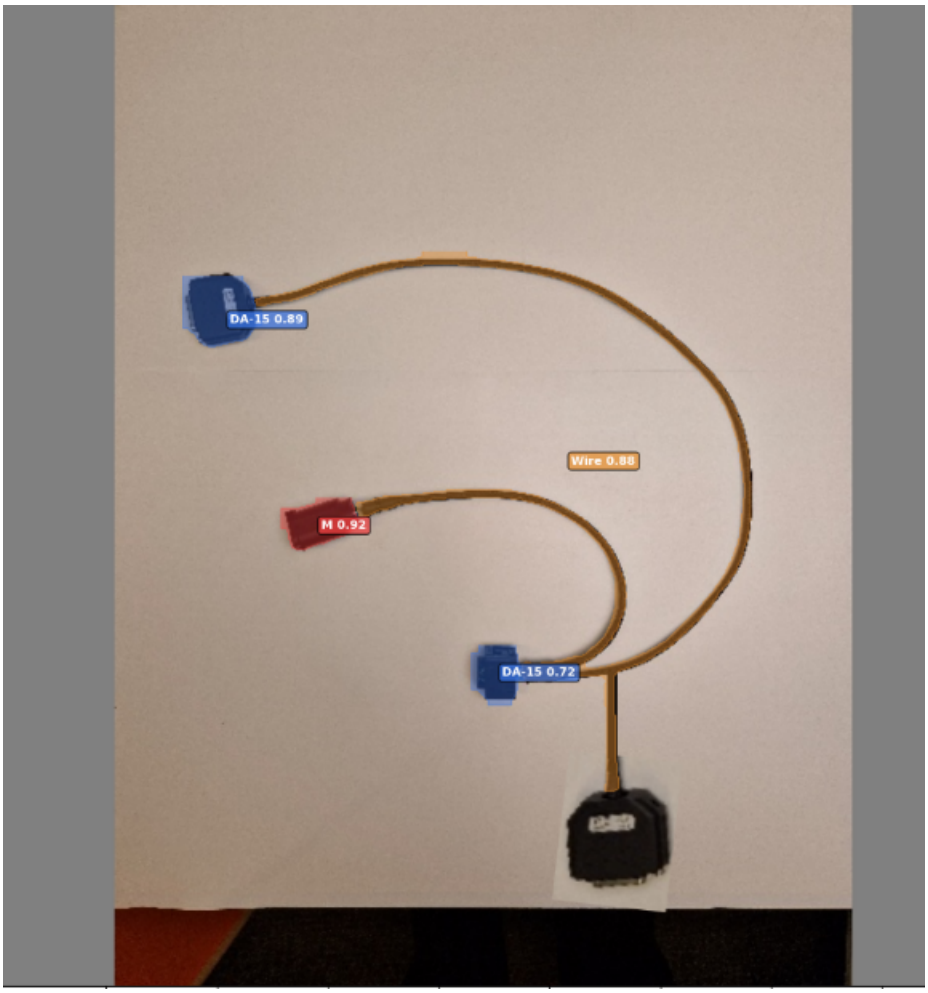


Figure 7.9: Results from Model 4 on an artificial harness

7.3 Skeletonization

The skeletonization method receives the segmentation output. The following section demonstrates the skeletonization method, the blue dots are endpoints of wire segments while the green are junctions / branch points. Figure 7.10 are sketched in Figma. Figure 7.11 is also sketched in Figma and demonstrates the skeletonization method:

- Step 1: Receive the wire mask
- Step 2: Remove shorter branches (caused by bad segmentation)
- Step 3: Ensure connectivity of the wire mask segments.

Finally Figure 7.12 visualizes the extracted graph from the skeletonization on top of an image from D1. The grey line between the endpoints and junctions are only visualization of how they are connected not the actual path of the wire.

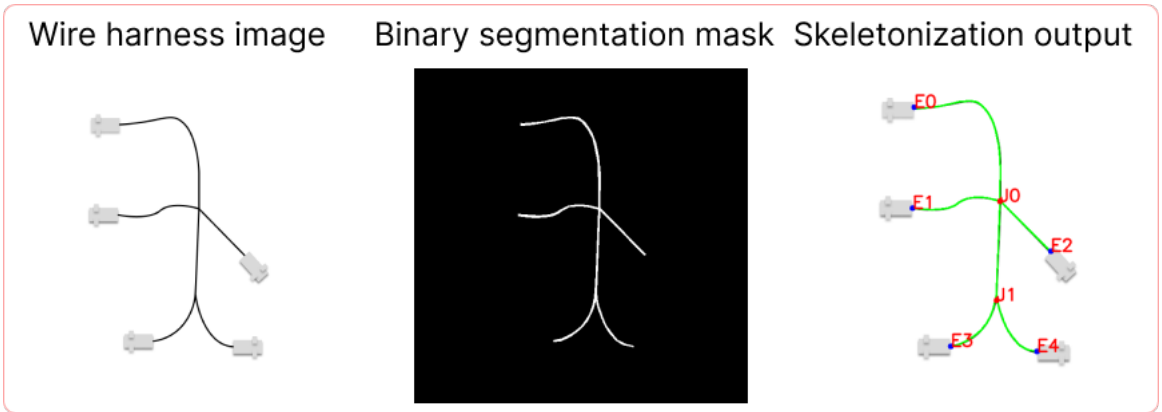


Figure 7.10: Skeletonization process, from sketched input image

Figure 7.11: Demonstrating the skeletonization mask pre processing

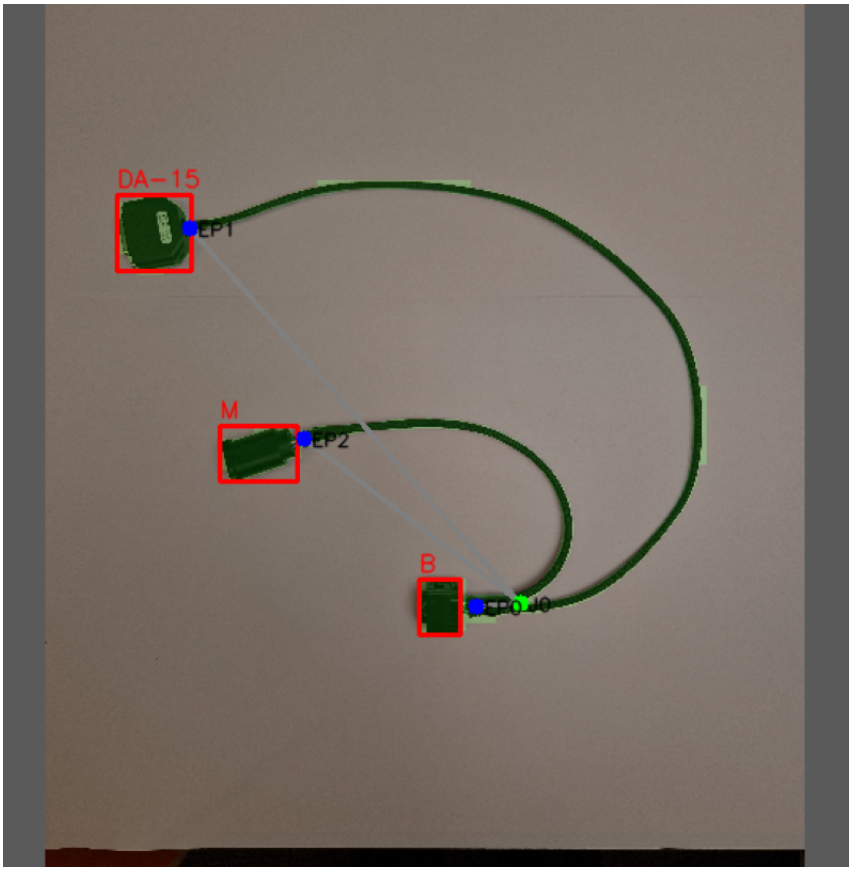


Figure 7.12: Topology prediction after skeletonization visualized on a wire harness from D1

The extracted graph is then saved in JSON format, below shows the topology generated from 7.12. The graph has one junction, three endpoints and three edges, all edges also include the length of each wire, presented in pixels.


```
{
  "metadata": {
    "created_at": "2025-04-23T13:58:02.741434",
    "total_images": 1
  },
  "images": {
    "20250218_141058.png": {
      "metadata": {
        "image_path": "test_images_resized/20250218_141058.png",
        "timestamp": "2025-05-10T14:08:39.698627"
      },
      "graph": {
        "nodes": {
          "J0": {
            "type": "junction",
            "position": [
              0.59375,
              0.6765625
            ],
            "label": "J0"
          },
          "EP0": {
            "type": "endpoint",
            "position": [
              0.55,
              0.6796875
            ],
            "label": "B"
          },
          "EP1": {
            "type": "endpoint",
            "position": [
              0.2671875,
              0.30625
            ],
            "label": "DA-15"
          },
          "EP2": {
            "type": "endpoint",
            "position": [
              0.3796875,
              0.5140625
            ],
            "label": "M"
          }
        },
        "edges": [
          {
            "from": "J0",
            "to": "EP0",
            "length": 616.8427124746208
```

```
            },
            {
              "from": "J0",
              "to": "EP1",
              "length": 275.3086578651021
            },
            {
              "from": "J0",
              "to": "EP2",
              "length": 30.48528137423857
            }
          ]
        },
        "skeleton_stats": {
          "total_pixels": 806,
          "non_zero_pixels": 806
        }
      }
    }
}
```


7.4 Classification

Table 7.6 demonstrates the graph convolutional network when tested on a JSON file with correctly structured graphs for all test graphs from D1&D2. In comparison with Table 7.7 that is tested on a JSON file generated from the best-performing segmentation model, Model 3.

Model	Dataset	Precision
1	D1 & D2	**1.0**

Table 7.6: Results from GCN model training

Model	Dataset	Precision
1	D1 & D2	**0.837**

Table 7.7: Precision of GCN on segmentation results

8

Discussion

The systematic literature review highlighted both strengths and weaknesses of previous studies within the wire harness community. They focus mainly on different areas such as extracting topological information of a harness, feature point prediction, semantic segmentation, and classification. From the SLR it is apparent that there are no previous studies that focus on the entire pipeline of extracting a graph representation to then classify the wire harness against known harness. [4], [2], [12] focuses on segmentation for DLOs and with a skeletonization technique to handle overlapping wires and focuses mainly on achieving a fast inference time. [20] work of evaluating architecture best suited to segment a wire harness shows some strengths of segmentation. However, compared to what this study tries to achieve, developing a generalized pipeline that can classify multiple different wire harnesses without any artificial markers. Their study only test their models on one wire harness from a car door with 9 different components to classify which components are present with the help of artificial markers.

The feature extraction papers [33], [22], [1] managed to extract features such as endpoints, junctions, and connectors. The main idea for their feature point extraction is to predict keypoints for further down-the-line robotic gripping points. For classification of a harness and accurate wire structures, these feature points would need to be interpolated like [23] does, which introduces some uncertainty of estimating wire structure.

[23] and [36] work with topological extraction and matching is promising for extracting a topology of a wire harness. After testing this approach in the experimental study, some flaws with this approach have emerged. For large wire segments with large overlapping bounding boxes, the keypoint predictions perform poorly. For a more accurate wire structure estimation of the interpolation method in [23] the model could be updated to predict more keypoints e.g 1000 for each wire segment. However, this comes with a computational cost and goes towards the computational cost of segmentation when increasing the number of keypoints.

The results of the experimental study demonstrate the strength of a segmentation pipeline together with a skeletonization method to extract a graph representation that can be classified with a GCN. Compared to pose estimation models where it was very difficult to achieve good results, this also aligns with the work of [33], which mentions the issue of large background noise around the bounding boxes for keypoint prediction. Segmenting the whole wire harness instead of just a certain

amount of keypoint interpolated with a spline method enables the ability to extract a more accurate length measurement of each wire segment, this is a critical feature for harnesses that look alike but differ in wire lengths. Even the best pose estimation model from [23] lacks accurate wire curvature predictions, as seen in Figure 7.2. There are limitations of the proposed segmentation solution aswell such as overlapping wires. For the skeletonization method to work correctly, it requires a segmentation without overlap as input. There are other solutions that solves this; [4] uses a minimum bending energy solution to get a prediction of the topology for overlapping wires.

The early stages of testing the segmentation model without synthetic data gave very poor segmentation when tested on images with backgrounds different than D1&D2. In order to make the model generalize better and learn to extract a wire harness from different backgrounds, the copy-paste method [38] was implemented. This improved the segmentation model ability to extract wire harnesses from various backgrounds. With the copy-paste method and rotation of wire harness also greatly increase the size of the dataset, by a multiplier of 32, which improved the accuracy of the segmentation model.

A solution involving keypoint detection works well for harnesses with short wire segments as can be seen in Figure 7.2. For harnesses with long wire segments the keypoint detection however works much worse, presented in Figure 7.3, the keypoints for the short wire follow the curvature of the wire. The second wire is much longer and the model predicted zero of the five keypoints correctly for this wire. The reason for this is probably because the overlapping bounding boxes there are multiple wire segments in the same bounding box and most of the bounding box content is background noise. D2 dataset from [23] is laid out flat and the wires are pretty straight. The D1 dataset often contains a longer wire that is rotated around the entire harness.

The GCN has a 100% accuracy, this however is not so extraordinary because the harnesses has maximum of one overlapping connector. So, the edge information has probably very little impact on the prediction of this model. If the prediction of the segmentation model is correct, there can only be one wire harness in the validation file with those three connectors. A more accurate result would be finding more wire harnesses that include the same connectors but different topologies. The GCN outputs a confidence score for each prediction and with the help of the validation file, a list of completeness for best-matched harness is returned. This allows for a deeper insight into the classification and can be used further down the line for quality control or error detection at the wire harness assembly stations.

The following tables compare the results of the different parts of the proposed segmentation pipeline for graph extraction and classification. Each table compares a specific part of the pipeline and compares against other papers that focus on the same area. The first table, Table 8.1 evaluates segmentation model 3 (best one) on dataset D1&D2 against [20], [20] did not open source their image of the car door so

comparison on the same dataset is not possible. Both evaluate deep learning semantic segmentation models, and as can be seen from the table [20] performs better on both IoU and precision while our model (our model is refereed to best performing segmentation model, Model 3) has a higher mAP@50. This is believed to be because their model only focuses on one wire harness with artificial markers, tied in a fixed position with clips. That our model has a higher mAP@50 indicates that with a lower threshold for the IoU, its has more coarse but correct detections than [20].

Model	**Dataset**	**IoU**	**Precision**	**mAP@50**
Our (model 3)	D1&D2	0.787	0.827	**0.973**
[20]	Proprietary dataset (car door)	**0.862**	**1**	0.89

Table 8.1: Comparison of wire segmentation results

Table 8.2 compares the segmentation only for the class: Wire, against the primary papers that focus on segmentation of DLOs. The IoU and precision for our model beats both [4] and [2]. While the Dice score is higher for [4] probably because their wires are thinner than the once from D1&D2 where the DICE score is a more stable metric than IoU. As seen in Table 7.5 pre-training on the electrical dataset did not manage to capture the larger black wire segments from D1&D2 at all. After trained on D1&D2 it performed very lightly better than not pre-training on thin electrical wires. Trying to freeze the initial layers of the model after train on electrical wires then D1&D2 it manage to segment the wire, but perform worse when trying to predict the connectors as seen in Figure 7.9.

Model	**Dataset**	**Dice**	**IoU**	**Precision**
Our (model 3)	D1&D2	0.950	**0.904**	**0.904**
[4]	Electrical dataset	**0.973**	0.761	0.862
[2]	Electrical dataset	-	0.89	-

Table 8.2: Comparison of wire segmentation results for only wires

Finally, Table 8.3 compares our classification model to the primary papers from the SLR that focuses on classification of wire harnesses. [23] solutions are also included because it focuses on graph extraction. The precision of the GCN in the proposed solution is the same as the precision of the segmentation model. This is because of the validation file, when the GCN makes a prediction its compared against the validation file, as mentioned in section 6.4.2. The classification model returns a list of errors if the extracted graph do not match any of the known harnesses in the validation file. So, the GCN precision is highly dependant on the segmentation input, this is why [23] is comparable.

The best-performing model in Table 8.3 is [20], which manages to obtain perfect precision. However, as mentioned earlier, they only evaluate their model on one harness. Similarly, [11] also only focus on one harness and tries to classify four

different branches of one harness. This makes the most interesting comparison, our model compared to [23] which has also been trained on D2, along with a synthetic dataset. Our proposed solution out performs their custom SPS developed model, their YOLO-pose model, however, beats ours; this is believed to be because of the short wire segments and limited overlapping bounding boxes in their dataset.

Model	**Dataset**	**Precision**
Our (model 3)	D1&D2	0.837
SPS [23]	D2 & Synthetic	0.825
YOLO [23]	D2 & Synthetic	0.898
[11]	Their proprietary dataset	0.967
[20]	Their proprietary dataset	**1**

Table 8.3: Comparison of classification of a wire harness

8.1 What kind of visual features can be used to represent a wire harness in an abstract way?

The primary papers identify different features that can be used to describe the harness in an abstract way. Different keypoints on the wire harnesses such as endpoints and junctions which can be transformed into a graph. Visual differences like color, diameter, and length of the wire and what connectors/terminals that are present in the wire harness aswell as geometric features like spline curvature.

8.2 What kind of network architectures are suitable for extracting topological information from a wire harness image?

Several different network architectures can extract topological information from an image of a DLO. Direct topology method with Swin transformer for keypoint prediction. Semantic segmentation with YOLOv11, U-net, LinkNet, FPN manages to extract the segmentation of the wire. Once segmentation is achieved, the skeletonization method developed in this project can transform such an input mask to a topological structure. Pose estimation models are good for extracting topological information, but for an accurate graph representation, each edge should have some length information, in this case the length of the wires they underperform. In this study, a YOLOv11 segmentation model outperformed keypoint-based pose models, especially in handling long wire segments and background noise. These results support the use of segmentation-first pipeline along with a skeletonization to extract topological information.

8.3 How can you classify a wire harness to previously known harnesses?

The combination of a segmentation model, skeletonization method, and GCN-based graph classification proved effective for non-overlapping harnesses. However, future work involving techniques to handle overlapping wires must be implemented to make model handle more real word scenarios. The GCN reached a 100% accuracy; however, this is very limited to different topologies and similar connectors across harnesses. The proposed solution can easily generalize to new harnesses as proved in figure 7.8. The classification also returns very detailed results when the GCN prediction is validated with a validation file, in order to return the best match along with the eventual mismatching labels or edges.

9
Conclusion

From this study it is evident that a computer vision model good enough and capable of automated wire harness assembly is still missing. This study does not provide a plug-and-play solution for automating the wire harness assembly either. It shows promising steps though; evaluating different architectures for extracting a graph representation and classification of wire harnesses. It also shows that keypoint detection to extract a graph representation can achieve high accuracy. This, however does not yield the same feature-rich information as the segmentation pipeline. A segmentation pipeline with a skeletonization method for graph representation results in a feature-rich graph representation that can easily be classified against known harnesses. The skeletonization method for extracting the topological features is dependent on the quality of the wire segmentation, so future work will try to improve the segmentation model as much as possible. The GCN along with a validation file makes it very easy to add new harnesses for applications involving wire harness classification. A new harness is introduced by just adding the harness to the validation file; while also training the GCN on the new harness structure. The YOLOv11 model does not need additional training for new harnesses as long as there are no new connectors introduced. The solution proposed in this study is not able to correctly extract a graph representation of harnesses with overlapping wires. This will be further worked on and a similar solution to [4], where a minimum bending energy method was implemented to handle overlapping wires, will be explored. Inference time was not evaluated during this study, this is another area that can be further worked on. The pipeline developed in this study can be used in areas such as quality control and error detection in manufacturing, contributing to the vision of automating the wire harness industry.

Bibliography

[1] Alessio Caporali, Kevin Galassi, and Gianluca Palli. Deformable linear objects 3d shape estimation and tracking from multiple 2d views. *IEEE Robotics and Automation Letters*, 8(6):3852–3859, 2023.

[2] Alessio Caporali, Kevin Galassi, Riccardo Zanella, and Gianluca Palli. Fastdlo: Fast deformable linear objects instance segmentation. *IEEE Robotics and Automation Letters*, 7(4):9075–9082, 2022.

[3] Abhishek Chaurasia and Eugenio Culurciello. Linknet: Exploiting encoder representations for efficient semantic segmentation. *CoRR*, abs/1707.03718, 2017.

[4] Andrew Choi, Dezhong Tong, Brian Park, Demetri Terzopoulos, Jungseock Joo, and Mohammad Khalid Jawed. mbest: Realtime deformable linear object detection through minimal bending energy skeleton pixel traversals. *IEEE Robotics and Automation Letters*, 8(8):4863–4870, 2023.

[5] Pao-Nan Chou. A comparison study of impact factor in web of science and scopus databases for engineering education and educational technology journals. *Issues in Informing Science and Information Technology*, 9:187–194, 01 2012.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE, 2009.

[7] Sofia Espana, Juan Sanchez, Carlos Saldarriaga, Bryan Puruncajas, Sungwon Seo, Hyungpil Moon, and Francisco Yumbla. Grip points generation on motorcycle wire harnesses main branch based on computer vision and clustering. In *2024 10th International Conference on Automation, Robotics and Applications (ICARA)*, pages 117–122, 2024.

[8] Trisha Greenhalgh and Richard Peacock. Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources. *BMJ*, 331(7524):1064–1065, 2005.

[9] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354 – 377, 2018. Cited by: 4866.

[10] Martin Kaloev and Georgi Krastev. Experiments focused on exploration in deep reinforcement learning. In *2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 351–355, 2021.

[11] Piotr Kicki, Michał Bednarek, Paweł Lembicz, Grzegorz Mierzwiak, Amadeusz Szymko, Marek Kraft, and Krzysztof Walas. Tell me, what do you see?—interpretable classification of wiring harness branches with deep neural networks. *Sensors*, 21(13), 2021.

[12] Piotr Kicki, Amadeusz Szymko, and Krzysztof Walas. Dloftbs – fast tracking of deformable linear objects with b-splines, 2023.

[13] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[14] Barbara Kitchenham. Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004):1–26, 2004.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. volume 2, page 1097 – 1105, 2012. Cited by: 85521.

[16] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. *CoRR*, abs/1612.03144, 2016.

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision (ECCV)*, pages 740–755. Springer, 2014.

[18] Pavan Kumar Mallapragada, Rong Jin, Anil K. Jain, and Yi Liu. Semiboost: Boosting for semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):2000–2014, 2009.

[19] Alberto Martín-Martín, Mike Thelwall, Enrique Orduna-Malea, and Emilio Delgado López-Cózar. Google scholar, microsoft academic, scopus, dimensions, web of science, and opencitations' coci: a multidisciplinary comparison of coverage via citations. *Scientometrics*, 126(1):871–906, 2021.

[20] Huong Giang Nguyen and Jörg Franke. Deep learning-based optical inspection of rigid and deformable linear objects in wiring harnesses. *Procedia CIRP*, 104:1765–1770, 2021. 54th CIRP CMS 2021 - Towards Digitalized Manufacturing 4.0.

[21] Khang Nguyen, Luu T. V. Ngo, Kiet T. V. Huynh, and Nguyen Thanh Nam. Empirical study one-stage object detection methods for robocup small size league. In *2022 9th NAFOSTED Conference on Information and Computer Science (NICS)*, pages 264–268, 2022.

[22] Thong Phi Nguyen and Jonghun Yoon. A novel vision-based method for 3d profile extraction of wire harness in robotized assembly process. *Journal of Manufacturing Systems*, 61:365–374, 2021.

[23] Shengzhe Ouyang, Manuel Zürn, Lukas Zeh, Armin Lechler, and Alexander Verl. Topology prediction of branched deformable linear objects using deep learning. *IEEE Access*, 12:194399–194411, 2024.

[24] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee and. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77, 2007.

[25] P Prajwal, D Prajwal, D H Harish, R Gajanana, B S Jayasri, and S. Lokesh. Object detection in self driving cars using deep learning. In *2021 International*

Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), pages 1–7, 2021.

[26] Aishwarya Prakash and Shweta Chauhan. A comprehensive survey of trending tools and techniques in deep learning. In *2023 International Conference on Disruptive Technologies (ICDT)*, pages 289–292, 2023.

[27] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[29] Anurag Sharma, Amanpreet Kaur, and Amit Semwal. Supervised and unsupervised prediction application of machine learning. In *2022 International Conference on Cyber Resilience (ICCR)*, pages 1–5, 2022.

[30] Hao Wang and Björn Johansson. Deep learning-based connector detection for robotized assembly of automotive wire harnesses. pages 1–8, 2023.

[31] Hao Wang, Omkar Salunkhe, Walter Quadrini, Dan Lämkull, Fredrik Ore, Mélanie Despeisse, Luca Fumagalli, Johan Stahre, and Björn Johansson. A systematic literature review of computer vision applications in robotized wire harness assembly. *Advanced Engineering Informatics*, 62:102596, 2024.

[32] Ye Wang, Xuewu Wang, Sanyan Chen, and Xingsheng Gu. Integrated optimization of multi-station multi-robot assembly welding line: Application for automotive industry. *Expert Systems with Applications*, 267, 2025. Cited by: 0.

[33] Yuxuan Yang, Johannes A. Stork, and Todor Stoyanov. Tracking branched deformable linear objects using particle filtering on depth images. In *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, pages 912–919, 2024.

[34] Riccardo Zanella, Alessio Caporali, Kalyan Tadaka, Daniele De Gregorio, and Gianluca Palli. Auto-generated wires dataset for semantic segmentation with domain-independence. In *2021 International Conference on Computer, Control and Robotics (ICCCR)*, pages 292–298, 2021.

[35] Riccardo Zanella, Alessio Caporali, Kalyan Tadaka, Daniele De Gregorio, and Gianluca Palli. Auto-generated wires dataset for semantic segmentation with domain-independence. In *2021 International Conference on Computer, Control and Robotics (ICCCR)*, pages 292–298, 2021.

[36] Manuel Zürn, Annika Kienzlen, Lars Klingel, Armin Lechler, Alexander Verl, Shiyi Ren, and Weiliang Xu. Deep learning-based instance segmentation for feature extraction of branched deformable linear objects for robotic manipulation. In *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, pages 1–6, 2023.

[37] Manuel Zürn, Markus Wnuk, Armin Lechler, and Alexander Verl. Topology matching of branched deformable linear objects. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7097–7103, 2023.

[38] Bare Luka Žagar, Alessio Caporali, Amadeusz Szymko, Piotr Kicki, Krzysztof Walas, Gianluca Palli, and Alois C Knoll. Copy and paste augmentation for deformable wiring harness bags segmentation. In *2023 IEEE/ASME Interna-*

tional Conference on Advanced Intelligent Mechatronics (AIM), pages 721–726, 2023.

A

Appendix 1

A.1 Json topology Appendix


```
{
  "metadata": {
    "created_at": "2025-04-23T13:58:02.741434",
    "total_images": 1
  },
  "images": {
    "20250218_141058.png": {
      "metadata": {
        "image_path": "test_images_resized/20250218_141058.
           png",
        "timestamp": "2025-04-23T14:08:39.698627"
      },
      "graph": {
        "nodes": {
          "J0": {
            "type": "junction",
            "position": [
              0.59375,
              0.6765625
            ],
            "label": "J0"
          },
          "EP0": {
            "type": "endpoint",
            "position": [
              0.55,
              0.6796875
            ],
            "label": "B"
          },
          "EP1": {
            "type": "endpoint",
            "position": [
              0.2671875,
              0.30625
            ],
```

```
            "label": "DA-15"
          },
          "EP2": {
            "type": "endpoint",
            "position": [
              0.3796875,
              0.5140625
            ],
            "label": "M"
          }
        },
        "edges": [
          {
            "from": "J0",
            "to": "EP0",
            "length": 616.8427124746208
          },
          {
            "from": "J0",
            "to": "EP1",
            "length": 275.3086578651021
          },
          {
            "from": "J0",
            "to": "EP2",
            "length": 30.48528137423857
          }
        ]
      },
      "skeleton_stats": {
        "total_pixels": 806,
        "non_zero_pixels": 806
      }
    }
  }
}
```

Department of Industrial and Materials Science
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY