Template Matching-based Method to Detect Bridge Components in Point Clouds

Schatz Y., Domer B.
University of Applied Sciences and Arts Western Switzerland, Switzerland
yohann.schatz@hesge.ch

Abstract. Monitoring and maintenance activities for bridges benefit from the use of digital twins. A digital twin can be defined as a numerical model of an existing structure enhanced with real-time data, representing the actual state of an asset. The manual preparation of the underlying numerical model is considered to be time-consuming, consequently automating the Scan-to-BIM process is of particular interest. One major challenge of automation is to segment the built asset by automatically detecting its components. This contribution proposes a methodology for bridge component detection based on a template-matching algorithm. Tests showed the accuracy of the proposed solution, even for incomplete point clouds, without requiring extensive input datasets. Compared to other solutions, this approach shows potential for adaptation to a variety of bridge designs.

1. Introduction

Digital twins simulate the behavior and evolution of existing physical entities through digital models enhanced with real-time data from sensors or external sources. A detailed definition is provided by Tekinerdogan and Verdouw (Tekinerdogan and Verdouw, 2020). One promising application in AEC industry is the automated inspection, assessment and management of built assets (Truong-Hong and Lindenbergh, 2022; Yang et al., 2022). BIM models expressed through an interoperable format like IFC (buildingSMART, 2022) are good candidates to serve as an integrative basis.

Creating digital twins for bridges is of particular interest. When counting road bridges in Switzerland, there are more than 4500 requiring active monitoring and maintenance measures (OFROU, 2016). Only a minority is modeled digitally. However, such a model is needed as a basis for a digital twin approach for maintenance purposes. Modeling existing bridges without intelligent computational support is considered to be inefficient. As a consequence, researchers are proposing methods to automate this task and integrate maintenance activities (Opoku et al., 2021).

At present, 3D scanning is a fast and reliable solution to obtain digital information about the shape and other visual features of an existing bridge. The output is a collection of three-dimensional data points in space known as "point cloud" (Yang et al., 2022). The process of creating a BIM model from a point cloud is commonly referred to as "Scan-to-BIM" process.

The automated Scan-to-BIM process can be divided into the following steps: (1) detecting bridge components in point clouds (as labeled subsets), (2) generating solid geometries of components from their points and (3) structuring and enriching the digital model (according to information requirements). This paper focuses on the first step.

The literature proposes several methods to detect specific components in point clouds. They can be classified into two groups: algorithm-based and learning-based. Although promising, the application of learning-based methods is often limited by high computational costs or the absence of sufficient training data (Truong-Hong and Lindenbergh, 2022; Xia et al., 2022; Yang et al., 2022). Prominent examples among the numerous learning-based methods are PointNet, PointCNN and Dynamic Graph Convolutional Neural Network (DGCNN).

Algorithm-based methods can be further subdivided into bottom-up and top-down approaches. The bottom-up approach segments first the point cloud and classifies the obtained subsets, using expert knowledge. Segmentation relies on geometric features including surface normals, meshes, patches and nonuniform B-spline surfaces (Lu et al., 2019). Despite good results obtained in simple scenarios, this method is judged to be difficult to apply for real-world bridges, since it is sensitive to noise and outliers, lacks performance, and requires substantial effort to formulate classification rules. This is especially true for complex geometry (Xia et al., 2022; Yan et al., 2014).

The top-down approach, where segmentation and classification are performed simultaneously (i.e. segmentation is based on expert knowledge), appears to be more promising (Yang et al., 2022). Lu (Lu et al., 2019) developed a slicing algorithm for detecting components in point clouds obtained from scans of existing reinforced concrete bridges. It is proposed to separate deck and pier assemblies first. Then, pier caps and girders are detected and segmented, using surface normals and bounding box/density histograms, respectively. Although the extraction of the components showed to be very accurate, the method requires many input parameters and extensive expert knowledge related to the analyzed bridge type. This might complicate the adaptation to other bridge designs. In addition, only a fixed number of component types can be detected. Generally, the top-down approach is considered to be reliable but demanding in terms of preparation (i.e., manual preprocessing of the point cloud) and laborious to generalize (Xia et al., 2022; Yang et al., 2022).

This paper introduces a new variant of the top-down approach addressing the above-mentioned challenges. Based on the work of Lu (Lu et al., 2019) and Zhao (Zhao et al., 2019), the presented method is intended to be flexible (i.e., easily adaptable to various bridges) and straightforward (i.e., simple to configure and use).

2. Methodology

The novelty of the proposed method (Figure 1) is to employ a template matching algorithm to detect occurrence(s) of bridge components within multiple cross-sectional views of a bridge.

Template matching is a high-level machine vision technique used to find the location of a specific element in an image, called "source image". Required inputs are the source image and a "template image" containing the pattern to find in the source image (OpenCV, 2023).

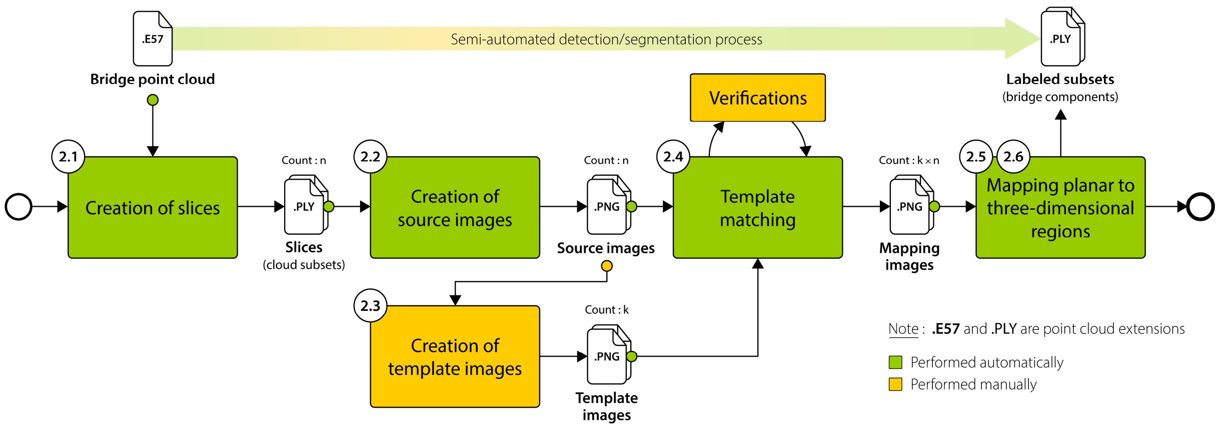

Figure 1 - Workflow for bridge component detection

In this work, source images are cross-sectional views of the bridge to be examined. Views are generated from slices (subsets) of the point cloud, each slice being linked to a different source image. Template images are cross-sectional views of components to be detected and defined manually by the user. When a component is located in a source image, points are extracted from the related slice and labeled. The process is repeated until all slices have been processed.

The different stages of the methodology presented in Figure 1 are detailed in the following.

2.1 Creation of slices

To create a set of representative cross-sectional views of a bridge (i.e., source images), it is first necessary to cut the point cloud longitudinally as if using a "virtual knife". Positions of cutting planes can be individually defined such that geometrical variations are taken into account. Source images will be created from slices obtained: for n slices, n different cross-sectional views of the bridge will be created, each related to a part of the bridge along its longitudinal axis.

Since processing a point cloud to create an image is computationally expensive, working with small subsets (slices) avoids high computational costs.

2.2 Creation of source images

Source images are raster images, i.e., they consist of a grid of square pixels. They are created by orthographic projection of voxelized point cloud slices (Figure 2).

First, the slice is rotated so that its longitudinal direction is parallel to $\vec{x}$. Then, the slice is voxelized and voxels are "flattened" along $\vec{x}$ to obtain a two-dimensional picture composed of pixels. Pixels have $\{u, v\}$ coordinates related to $\{y, z\}$ coordinates of the point cloud.

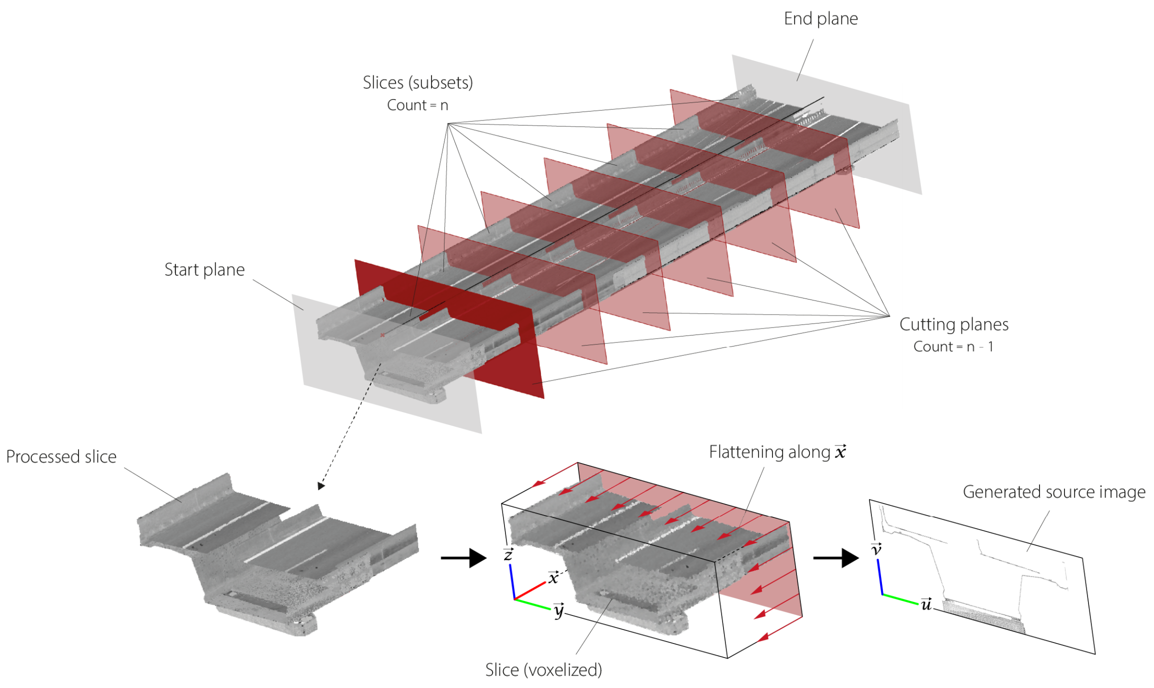

Figure 2 - Creation of source images

2.3 Creation of template images

Template images (Figure 3) are also raster images, created manually by a domain expert using a raster graphics editor such as GIMP. Each template illustrates a pattern, related to a component type (e.g., beam, box girder, guardrail, etc.), that will be used to check if occurrences of that component are present in the generated cross-sectional views of the bridge (i.e., source images). Ideally, template sets are defined for each bridge point cloud, considering all objects to be detected.

Template images can be created from scratch or derived from an existing image. Multiple template images can be defined for a single component type to account for variations in cross-sectional representations from one slice to another (Figure 3). Some components, such as piles, can be detected using a typical and recurrent pattern. In this case, many areas are matched to retrieve the entire object.

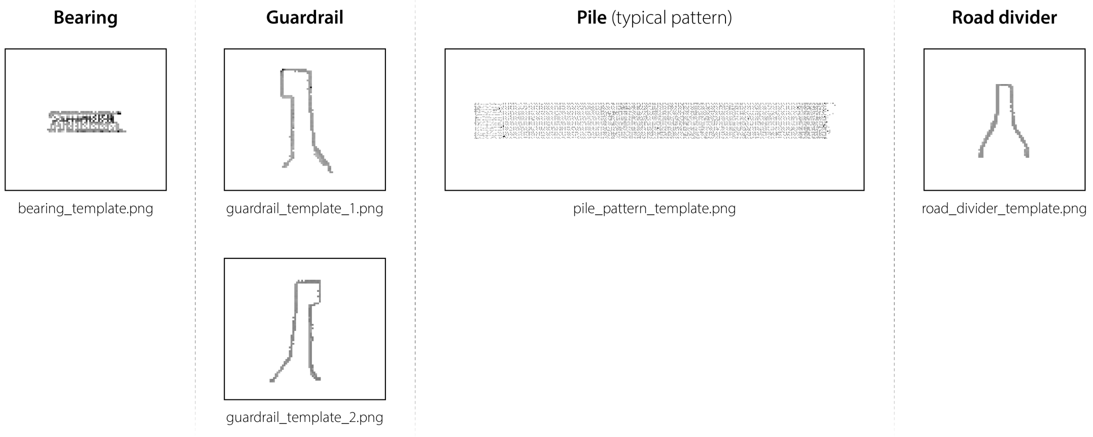

Figure 3 - Examples of template images

2.4 Template matching

Feature-based and area-based algorithms have been identified as candidates for the task of template matching (Swaroop and Sharma, 2016).

The feature-based approach relies on local features (such as interest points, curves, etc.) of images to find matches. Since features are invariant to scale, rotation, translation and intensity, the detection capacity is not very sensitive to changes in the appearance of components in the different source images. However, this method is inefficient when input images do not have strong features (e.g., low resolution or level of detail) (Swaroop and Sharma, 2016).

The area-based approach is iterative: at the beginning, the template image is positioned on the top-left corner of the source image and the similarity between the template and the overlapped area of the source image is evaluated using a comparison method (such as squared cross-correlation) and a "matching score" is associated to the current position. In the next iteration, the template is moved one pixel to the right (or downwards) and a new score is calculated for this position. The process is repeated until the template has moved to the bottom right corner. Finally, the position(s) that reach a threshold score is (are) extracted. The OpenCV library documentation (OpenCV, 2023) details the process and the various associated comparison methods.

This approach is robust when components do not change in size or orientation. Creating rotated or resized versions of a template is a straightforward solution to handle geometrical variations, but might significantly reduce the performance of the algorithm.

Regardless of the chosen approach, the expected result for each template matching execution is an image called "mapping image", in which matching regions are colored in red (Figure 4).

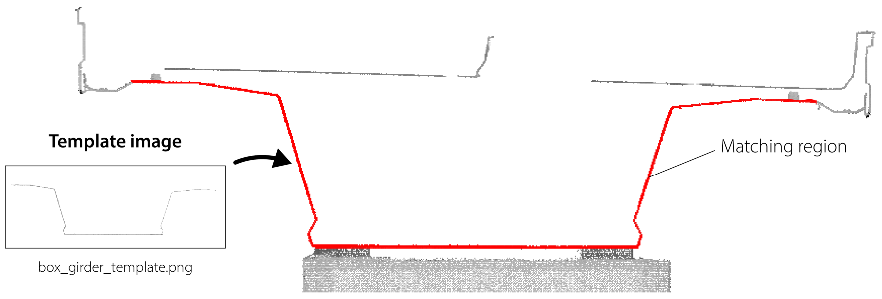

Figure 4 - Example of mapping image (matching result)

2.5 Mapping planar to three-dimensional regions

Red-colored (planar) regions of the mapping image must be mapped to three-dimensional regions of the point cloud to label the corresponding points.

The proposed methodology (Figure 5) is naively the inverse of the "flattening" operation presented in Figure 2: planar regions are extruded along $\vec{x}$ to obtain three-dimensional bounding boxes enclosing the points to be labeled. Extrusion length is equal to the longitudinal length of the slice.

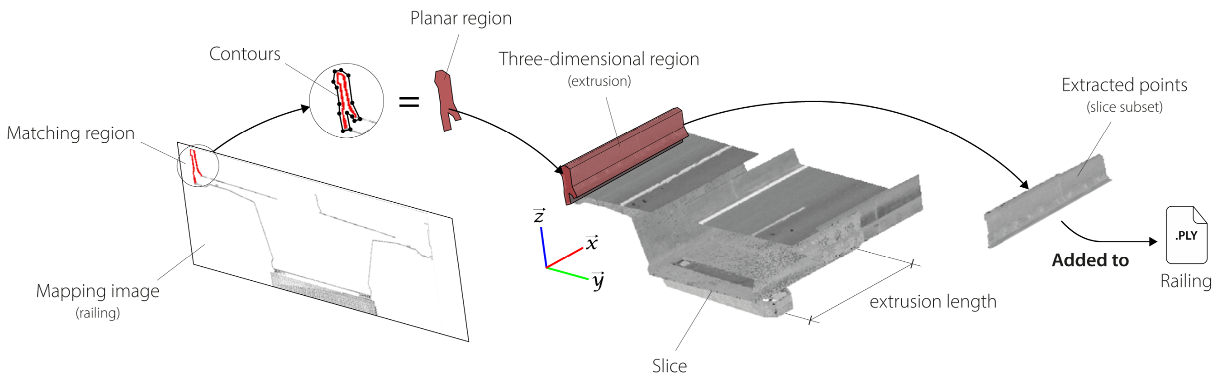

Figure 5 - Mapping planar to three-dimensional regions

2.6 Points extraction and labeling

Finally, points inside bounding boxes are extracted and integrated into a subset labeled with the name of the component type (Figure 5).

If this component type is found in other slices as well, extracted points will be added to the same labeled subset. Finally, considering p types of components to detect, p labeled subsets are obtained, containing points from all slices where the respective component was found.

3. Case study

The procedure presented before has been applied to detect and extract structural and non-structural components from point clouds of two bridges located in Switzerland (Figure 6).

A set of algorithms combining numpy (numerical computing), Open3D (3D data processing), OpenCV (computer vision) and pandas (data analysis) libraries for Python has been developed to generate source images (2.2), for template- (2.4) and region mapping (2.5) and the extraction of points (2.6).

Test bridges are curved box girder bridges, made of reinforced concrete. They have a length of 380 m (bridge 1) and 350 m (bridge 2). Although their geometry is relatively simple, a large number of voids and missing parts in their point clouds generated by 3D scans makes component detection challenging. This is a realistic scenario since environmental conditions and/or the employed survey method do not always allow the generation of clean and complete point clouds.

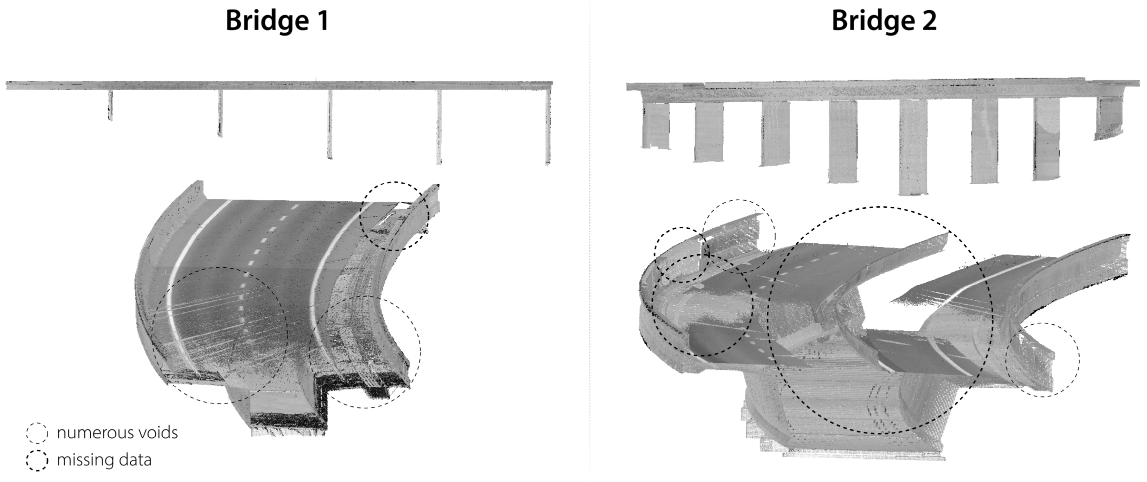

Figure 6 - Test bridges

3.1 Slices and templates images creation

Slices (.xyz files) are created using Rhinoceros 3D and Grasshopper (34 for Bridge 1, and 51 for Bridge 2). Some representative cross-sectional views of both bridges are generated from a set of slices (those with the fewest voids). These are edited in GIMP to manually produce template images (9 for Bridge 1, and 6 for Bridge 2). In the present case, the creation of templates took approximately 15 minutes per bridge. For both bridges, it is expected to extract the following components: bearings, box girders, drainpipes, guardrails, piles, road dividers and road surfaces.

3.2 Template matching algorithm selection and configuration

Tests were performed to determine the most suitable and robust template-matching approach given the incompleteness of the input point clouds (Figure 6). Feature-based matching algorithms yielded unsatisfactory results, likely due to the presence of noise and a too-low resolution of generated images. As expected, area-based matching algorithms produced more reliable results overall. This approach is therefore preferred and has been applied.

A major weakness of basic area-based matching algorithms is the increased risk of confusion (wrong matches) when templates become smaller since they are less likely to have particular and distinctive characteristics. To solve this issue, a new variant called "contextual area-based matching algorithm" has been implemented (Figure 7). The principle is to perform the sliding routine by using a "contextualized" version of the template (template + surrounding area) to filter and eliminate wrong candidates. Then, the area corresponding to the "original" template is retrieved from the previously matched region (in blue, Figure 7). The size of the surrounding area to consider is specific to the template and is defined by the user.

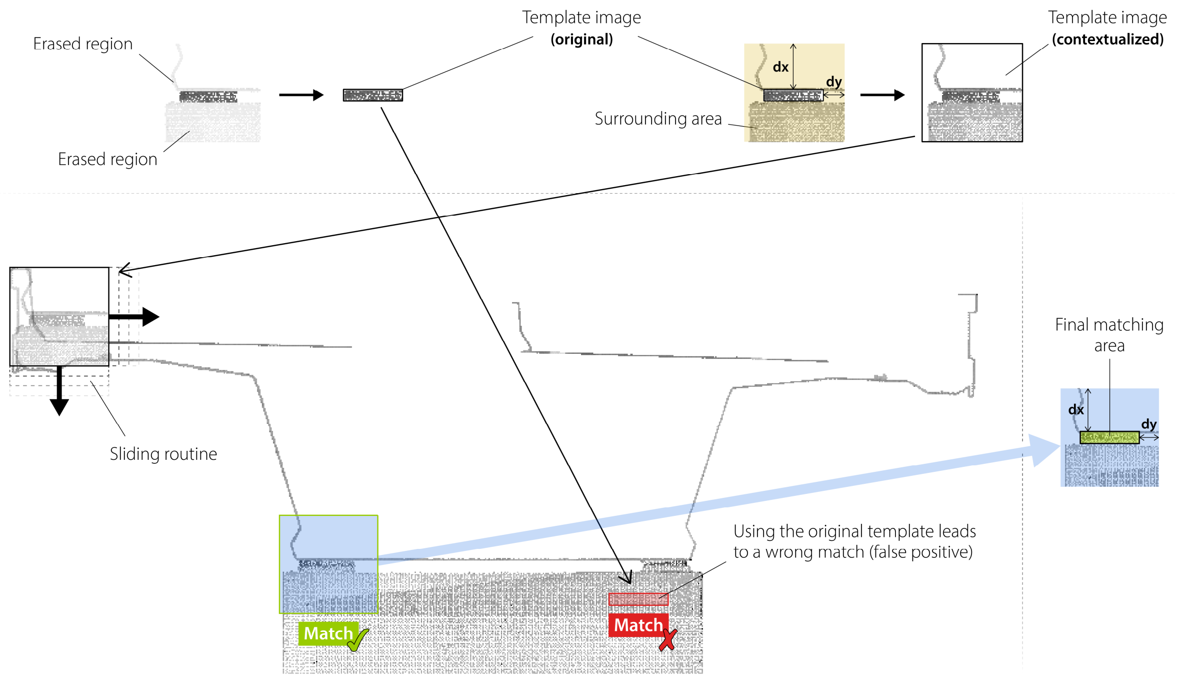

Figure 7 - Contextual area-based matching algorithm

The comparison method used is normalized cross-correlation (OpenCV, 2023), which returns for each candidate match a score between 0 (no match) and 1 (perfect match). Candidates are accepted or rejected based on the score, according to the decision intervals shown in Figure 8.

Lower and upper thresholds are defined by the user. If the score is in the "pending" interval, the user is prompted to check the matching result and make corrections if necessary. Finally, a tolerance value (defined by the user and specific to the template) is applied to retrieve only matches whose score is in the interval [*best score – tolerance, best score*] (Figure 8).

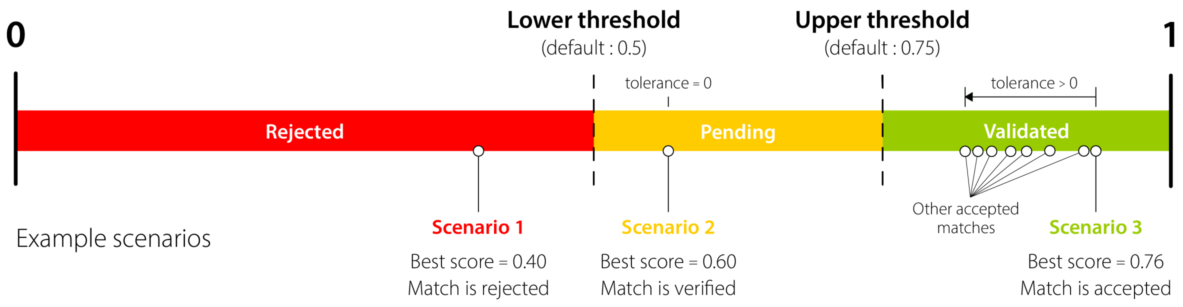

Figure 8 - Decision intervals

3.3 Results

Table 1 shows detailed results obtained with a standard machine (10-core CPU - 3.0 GHz clock rate - and 256 Gb RAM). Classification results for the second bridge (the most incomplete point cloud) are presented in Figure 9.

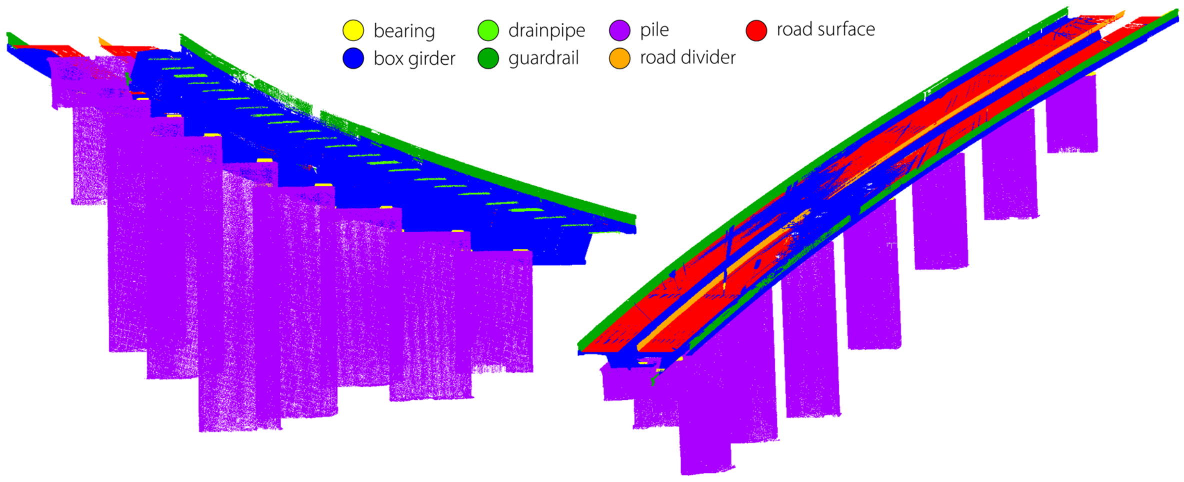

Figure 9 - Component detection results for Bridge 2

Table 1: Detailed results.

	Bridge 1	**Bridge 2**
Number of slices	35	46
Execution time *	313 s	911 s
Matching operations	204	414
True positives + negatives **	201 (98.5 %)	410 (99.0 %)
False positives + negatives **	3 (1.5 %)	4 (1.0 %)

* excluding the time taken by the user for manual verification/correction, when prompted.
** worst score obtained in the case where the user ignored manual verification requests (default threshold values).

Obtained results are encouraging, particularly for the second bridge whose scan is incomplete (Figure 9). All expected component types were detected with failure rates of 1.5 % for Bridge 1 and 1.0 % for Bridge 2. More precisely, there were no false negatives, and all potential false positives were identified (based on decision ranges) and submitted to the user for manual checking and correction. Wrong matches were mainly caused by confusing similar-looking components and could be minimized by refining templates and their parameters.

Contextualized area-based template matching has shown to be efficient in identifying components even with a limited number of points (Figure 10). This capability is however conditioned by the quality and appropriate configuration of templates.

It is assumed that the performance of the employed template matching algorithm will decline when components have a complicated geometry (e.g., variable section, sophisticated shape). Thus, the current solution seems to be appropriate for bridges having components with invariant geometry and where the point cloud is more or less incomplete.

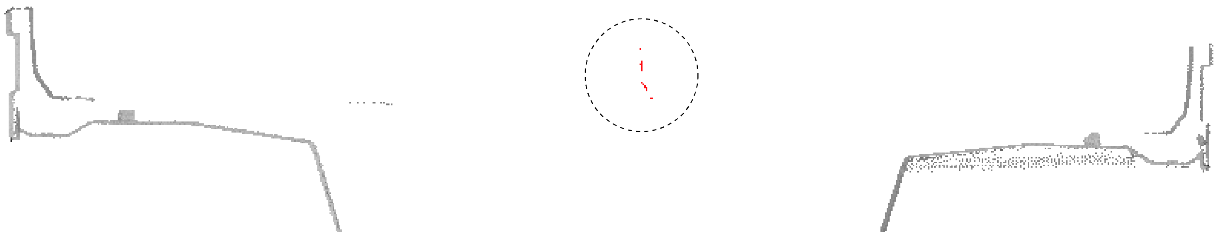

Figure 10 - Detection of a road divider from a limited number of points

The curvature of the examined bridges had no impact on component detection as the number of cutting planes was high enough to obtain "straight" slices, allowing to minimize "perspective effects" in cross-sections. However, the disadvantage of using more slices is that the process takes longer. In-depth analyses revealed that approximately 75% of execution time is spent on creating source images, generating mapping images and extracting points. Therefore, optimizing related algorithms bears a potential for time reduction.

4. Conclusion and future work

Detecting bridge components (semi-) automatically in point clouds is a challenging task of the Scan-to-BIM process. While most research efforts concentrate on machine learning-based methods, the potential of algorithm-based methods should not be neglected.

Among major algorithm-based solutions, those by Lu (Lu et al., 2019) introducing the principle of subdividing a bridge into slices and analyzing their features, were found to be very accurate for tested bridges. However, they suffer from complicated implementation and adaptation to other bridge designs than those used for testing.

The proposed solution requires subdividing the bridge as well. It employs a template-matching algorithm to detect bridge components in slices instead of employing complex features and rules. In contrast to learning-based methods, which need large training datasets, only a few bridge-specific templates are necessary to achieve segmentation.

A case study showed promising results in terms of detection accuracy, even for point clouds with noise, voids and missing parts. Relying on a semi-automatic approach, asking for user intervention when the score is situated in the "Pending" interval (Figure 8), limits the number of false matches. First studies suggested that adaptation to other bridge types could be less complex than for rule-based methods since the definition of templates requires only limited expert knowledge and effort.

The next step is to perform additional parametric tests to improve the proposed solution. Faced with the obvious limitations of area-based template matching algorithms, further tests using feature-based template matching algorithms will be carried out to propose an improved solution that better manages elements with complex geometries.

Acknowledgment

This research is supported by the University of Applied Sciences and Arts Western Switzerland in the framework of the project "BridgeTwin", co-directed by Haute École du Paysage, d'Ingénierie et d'Architecture de Genève (HEPIA) and Haute École d'Ingénierie et de Gestion du canton de Vaud (HEIG-VD). The authors would also like to thank the Swiss Federal Roads Office (ASTRA/OFROU) for discussions.

References

buildingSMART (2022). Industry Foundation Classes (IFC) - buildingSMART International. Available at: https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/, accessed October 2022.

Lu, R., Brilakis, I., Middleton, C.R. (2019). Detection of Structural Components in Point Clouds of Existing RC Bridges. Comput.-Aided Civ. Infrastruct. Eng. 34. 191–212. DOI: https://doi.org/10.1111/mice.12407.

OFROU (2016). Rapport sur l'état du réseau des routes nationales 2016.

OpenCV (2023). Template Matching in OpenCV. Available at: https://docs.opencv.org/3.4/de/da9/tutorial_template_matching.html , accessed March 2023.

Opoku, D.-G.J., Perera, S., Osei-Kyei, R., Rashidi, M. (2021). Digital twin application in the construction industry: A literature review. J. Build. Eng. 40. 102726. DOI: https://doi.org/10.1016/j.jobe.2021.102726.

Swaroop, P., Sharma, N. (2016). An Overview of Various Template Matching Methodologies in Image Processing. Int. J. Comput. Appl. 153. 8–14. DOI: https://doi.org/10.5120/ijca2016912165.

Tekinerdogan, B., Verdouw, C. (2020). Systems Architecture Design Pattern Catalog for Developing Digital Twins. Sensors 20. 5103. DOI: https://doi.org/10.3390/s20185103.

Truong-Hong, L., Lindenbergh, R. (2022). Automatically extracting surfaces of reinforced concrete bridges from terrestrial laser scanning point clouds. Autom. Constr. 135. 104127. DOI: https://doi.org/10.1016/j.autcon.2021.104127.

Xia, T., Yang, J., Chen, L. (2022). Automated semantic segmentation of bridge point cloud based on local descriptor and machine learning. Autom. Constr. 133. 103992. DOI: https://doi.org/10.1016/j.autcon.2021.103992.

Yan, J., Shan, J., Jiang, W. (2014). A global optimization approach to roof segmentation from airborne lidar point clouds. ISPRS J. Photogramm. Remote Sens. 94. 183–193. DOI: https://doi.org/10.1016/j.isprsjprs.2014.04.022.

Yang, X., del Rey Castillo, E., Zou, Y., Wotherspoon, L., Tan, Y. (2022). Automated semantic segmentation of bridge components from large-scale point clouds using a weighted superpoint graph. Autom. Constr. 142. 104519. DOI: https://doi.org/10.1016/j.autcon.2022.104519.

Zhao, Y.-P., Wu, H., Vela, P.A. (2019). Top-Down Partitioning of Reinforced Concrete Bridge Components. In: Computing in Civil Engineering 2019. Presented at the ASCE International Conference on Computing in Civil Engineering 2019, American Society of Civil Engineers. Atlanta, Georgia, pp. 275–283. DOI: https://doi.org/10.1061/9780784482445.035.