J geovis spat anal (2021) 5:10
https://doi.org/10.1007/s41651-021-00076-9

3D Reconstruction of Bridges from Airborne Laser Scanning Data and Cadastral Footprints

Steffen Goebbels[1]

Accepted: 15 February 2021

Abstract

The given paper describes a method for automatic 3D reconstruction of bridges from cadastral footprints and airborne laser scanning point clouds. The reconstructed bridges are used to enrich 3D city models. Unlike roofs, decks of bridges are typically smooth without ridge lines or step edges. Therefore, established methods for roof reconstruction are not suitable for bridges. The standard description language for semantic city models is CityGML. This specification of the Open Geospatial Consortium assumes that surfaces are composed of planar polygons. The approximation of smooth decks by planar polygons is achieved by using a medial axis tree. Instead of the medial axis of the footprint, a modified medial axis is computed that does not consider counter bearing edges. The resulting tree represents centerline connections between all counter bearing edges and, in conjunction with filtered height values of a point cloud, serves as the basis for approximation with polygons. In addition to modeling decks, superstructures such as pylons and cables are also derived from the point cloud. For this purpose, planes carrying many superstructure points are detected using the Random Sampling Consensus Algorithm (RANSAC). Images are generated by projecting points onto these planes. Then, image processing methods are used to find connected contours that are extruded to form 3D objects. The presented method was successfully applied to all bridges of two German cities as well as to large bridges built over the Rhine River.

Keywords 3D building reconstruction · CityGML · Airborne laser scanning · Point clouds

Introduction

For modeling digital twins of cities in 3D, CityGML is the standard description language of the Open Geospatial Consortium (see Gröger et al. (2012), Kutzner et al. (2020)). It is used not only for visualization purposes but also for multiple simulation tasks. It adds attributes and semantics to city objects. This allows the integration of data from different sources. Various applications of CityGML models are described in Biljecki et al. (2015).

Due to the huge number of objects to be modeled in a city, most CityGML models are largely computed automatically based on cadastral and laser scanning data. The focus so far has been mainly on the reconstruction of buildings based on roofs, cf. He (2015), Henn et al. (2013), and Perera and Maas (2012). However, roof plane detection techniques are not suitable for bridges. Unlike roofs, their decks have a smooth surface without ridge lines and step edges. Thus, requirements for automatic modeling of bridges are different from those for houses.

Bridges not only are important landmarks of cities, but are also needed for the modeling of traffic routes. Three-dimensional models of interchange bridges can assist with navigation (see Wang et al. (2018)).

In this paper, bridges are modeled based on cadastral data and airborne laser scanning point clouds. The cadastral data provide polygons that describe floating parts of bridges (cadastral footprint). In addition, lines representing counter bearings are used if available. The lines are matched with the polygons of the floating parts to mark counter bearing edges.

If no counter bearings are indicated in the cadastral data, a floating part edge is classified as a counter bearing if there are no significant elevation differences along the edge, i.e., if the edge connects the bridge to the ground.

Many bridges have a simple footprint that can be easily approximated by a rectangle, e.g., by calculating a principle

✉ Steffen Goebbels
Steffen.Goebbels@hsnr.de

[1] Faculty of Electrical Engineering and Computer Science, Institute for Pattern Recognition, Niederrhein University of Applied Sciences, 47805 Krefeld, Germany

Published online: 12 April 2021

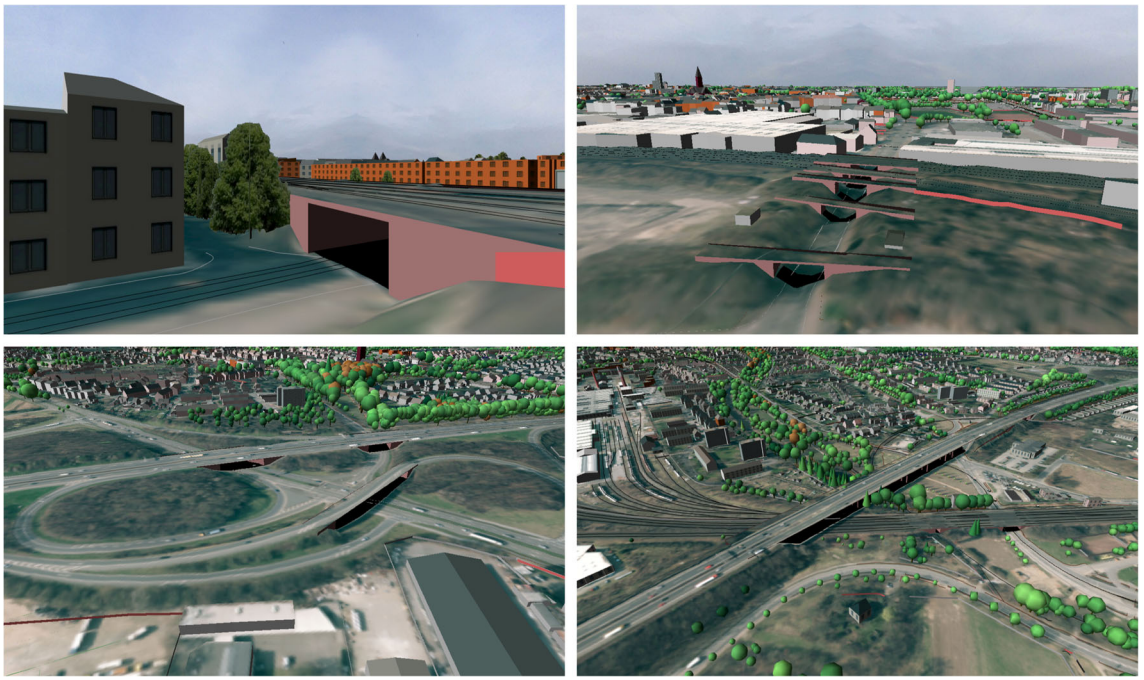

Fig. 1 Rectangular CityGML bridge models, which are textured with aerial images; floating parts were computed using principal component analysis. Then, floating parts were extended to close gaps between bridges and terrain model

component analysis of sampled points collected on the boundary of the floating part (see Fig. 1). However, the goal here is to create models with accurate boundaries of the floating part.

The Open Data Initiative of the German state of North Rhine-Westphalia publishes filtered laser scanning point clouds. Separate last-pulse clouds for bare earth and for aboveground structures are used. With respect to point clouds of structures above the ground, one could directly apply RANSAC Fischler and Bolles (1981) or the Hough transform to detect planar surfaces of bridge decks. Indeed, many bridges (e.g., railroad bridges) have a deck that can be modeled by a single plane obtained by RANSAC. This fits with the description language CityGML that requires surfaces to be composed of planar polygons. But bridge decks may not be planar. Attempts to reconstruct bridges that have non-constant slope using an algorithm for RANSAC-based roof reconstruction, see Goebbels and Pohle-Fröhlich (2016) produced poor results with bumpy surfaces and artificial step edges. Therefore, a different algorithm based on central lines of the footprint polygon is proposed. The edges of such polygonal central lines determine the direction of slope (gradient) so planar segments can be adjusted accordingly. Road centerlines, for example, have also been used previously in 3D road modeling, cf. Kada and Haala (2004) and Chen and Lo (2009).

Bridges can have more than two counter bearings (bifurcated layout, see Fig. 2) and their footprints can also have inner polygons (holes, e. g., bridges with traffic circles). To obtain central lines in this rather general setting, one can either compute a straight skeleton or a medial axis. A straight skeleton is computed by moving boundary edges to the interior of the polygon. The skeleton is formed by the intersection points with other edges. Skeletons are used, for example, in Cheng et al. (2015) to detect junctions. A medial axis consists of all points for which more than one nearest point exists on the footprint (see Fig. 3). Bridges with inner polygons (openings in the floating part) are not included in the test data used. If there are no inner polygons,

Fig. 2 Up and down ramps lead to branches: The bridge was reconstructed based on a tree obtained from a medial axis

both methods return a tree. Unfortunately, there may be no connections between the midpoints of counter bearing lines along edges of the tree. Therefore, if counter bearing lines are available from cadastral data, or if such lines can be computed using elevation differences, a modified medial axis is computed. For this purpose, the counter bearing edges are not considered as part of the footprint boundary (see Fig. 3, i.e., only points on boundary edges not belonging to a counter bearing are feasible candidates for nearest points). This creates a tree with leaves on counter bearing lines, covering paths connecting each pair of counter bearings. This method also helps to deal with bridges that are shorter than the width. Without excluding nearest points on counter bearings, a medial axis could be orthogonal to the direction of travel (cf. third bridge in Fig. 3).

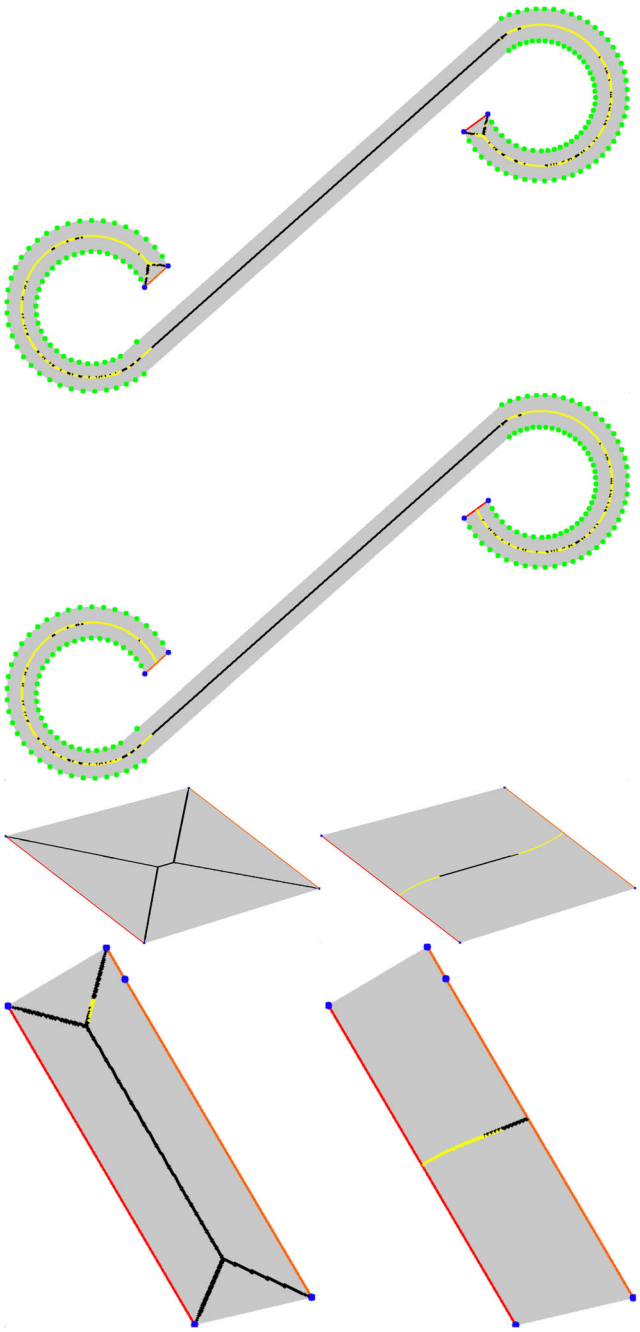

Fig. 3 Footprints of bridges: red lines and blue vertices represent counter bearings, yellow pixels represent points of the medial axis closest to at least one vertex, black points indicate points of the medial axis closest only to inner points of edges. In every second image, the medial axis is computed without considering edges and inner vertices of counter bearings

The next step is to compute height values (z-coordinates) for all points on the medial axis tree by low-pass filtering the point cloud data. To this end, we assume a smooth deck surface without step edges, i.e., pedestrian stairs are modeled as ramps. In any case, the resolution of the available point clouds is too low to recognize individual steps. Then, the tree edges are simplified using the Douglas-Peucker algorithm. The resulting tree is the basis for subdividing the deck into smaller polygons. Depending on planarity, these polygons might be divided and/or replaced by a triangle mesh. The method results in a polygon mesh adapted to the representation of a smooth deck. Sections 1–1 describe the decomposition of the deck into planar polygons in detail.

Finally, a 3D representation is created from deck polygons, and counter bearing structures are added. These are walls between the bare earth (from ground point clouds) and the deck. In some cases, cadastral data also provide footprints of pillars that in general are occluded by the deck in laser scanning point clouds. Pillar walls between ground and the deck are also added. Few bridges like suspension bridges have characteristic superstructures. Section 1 describes how a simple representation of superstructures can be derived directly from point clouds consisting of points above the ground. To this end, images are generated by projecting points to planes. Image processing methods are then applied to extract contours that are extruded into 3D solids.

Related Work

So far, there have been few attempts to automatically model CityGML bridges; see Wang et al. (2018) for an overview. For example, in Sithole and Vosselman (2003), the following assumptions are made:

1. A bridge touches bare earth at least on two sides.
2. Along its length, a bridge is raised above the ground.
3. Typically, a bridge is built longer than wide.

The first two observations can be used to detect positions of counter bearings when they are not indicated in cadastral data. However, in the data set for this study, many highway and railroad bridges are significantly wider than long. Therefore, the third assumption cannot be used.

Also in Sithole and Vosselman (2003), a scan-line algorithm is used to detect bridges in digital surface models.

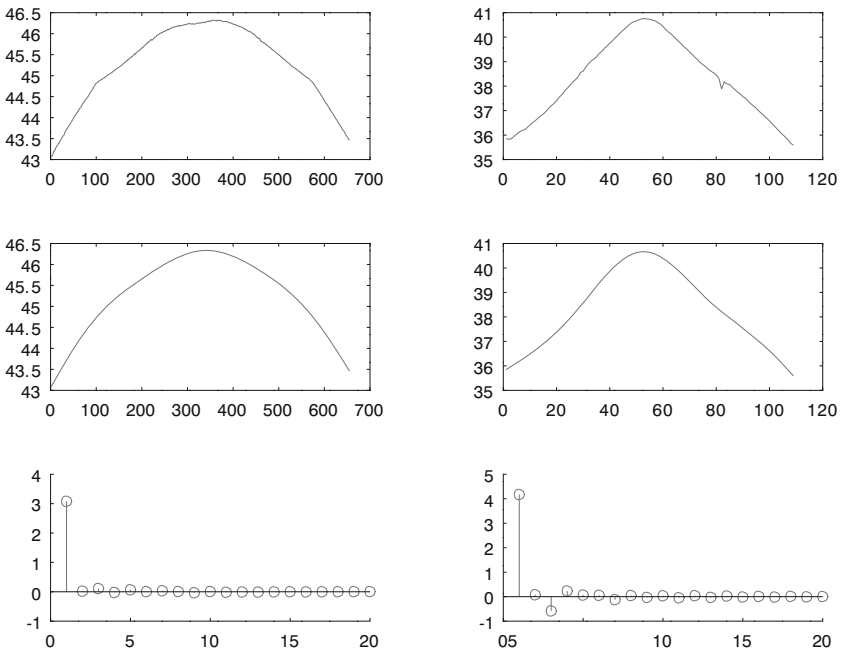

Fig. 4 Smoothing of z-coordinates of two bridge decks with low pass filtering. (The plots in the first column represent data from the self-anchored suspension bridge in Fig. 13. The second column belongs to the footbridge in Fig. 22, which also serves as an example in Figs. 3, 6, and 7.) The plotted functions f in the first row represent z-coordinates taken at sample points along the medial axis. The vertical axis is scaled in meters. The second row represents the result $\tilde{f}$ of low-pass filtering based on a Fourier partial sum (2) of sine functions. The first 20 Fourier coefficients $b_1, \ldots, b_{20}$ of the function g in (1) are shown in the last row

Fig. 5 The images show reconstructed bridges crossing at different levels and their integration into a 3D city model. Counter bearings are extruded to connect floating parts of the bridges with the ground

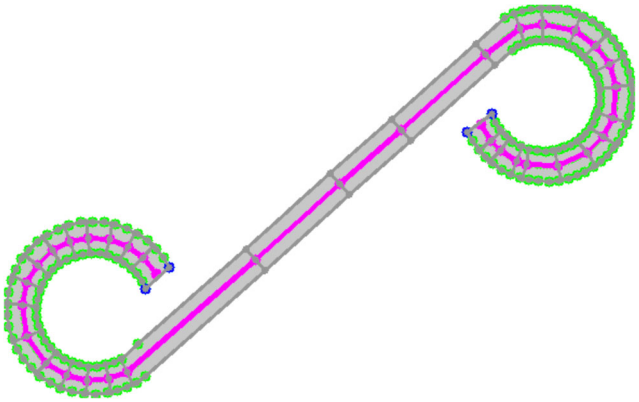

Fig. 6 Segments of the floating part obtained from the cadastral footprint and medial axis tree

The authors evaluate height discontinuities occurring on both sides of a bridge. In Sithole and Vosselman (2006) the detection of bridges with irregular footprints is described. In contrast to these approaches, the algorithms presented here do not only use point clouds but also cadastral data. Thus, it is known where bridges are located.

A deep learning approach to 3D model generation for suspension bridges from UAV images is described in Hu et al. (2020). It separates bridges from background, decomposes bridges into parts, and performs model driven detection of bridge parts by applying several deep neural networks. However, this sophisticated approach requires specific gathering of images while the simple algorithms in the given paper work with point clouds, which are freely available in North Rhine-Westphalia and other states. Therefore, the algorithms can be used to reconstruct a majority of bridges without manual intervention.

The 3D reconstruction of roads also includes to some extent the reconstruction of bridges. Here, the use of center lines is a common tool, cf. Chen and Lo (2009). However,

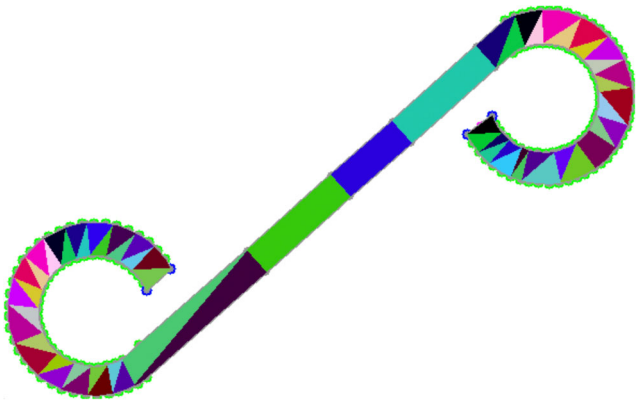

Fig. 7 Polygons obtained from the previously constructed tree

Fig. 8 The deck layout shows multiple polygons that have been split into two or three smaller polygons by introducing additional edges at cross-connection nodes

the approach taken here seems to be new: modeling the surface of a bridge based on planar polygons derived from the central line. It could be applied to 3D road reconstruction as well.

Modified Medial Axis as Skeleton

The proposed method is based on a modified medial axis computed on a raster image of a bridge, see Fig. 3. For each pixel, a set $\mathcal{M}$ of nearest points on the lines extending those boundary edges that do not belong to a counter bearing is determined. The set $\mathcal{N}$ of all near points consists of all footprint vertices (which are not inside a counter bearing edge) and of all points in $\mathcal{M}$ which are inside their own edge. If $\mathcal{N}$ contains at least two points with the same smallest distance to the pixel, then the pixel belongs to the medial axis. Since a raster representation is used, quantization errors must be expected. Therefore, distances that differ by at most a threshold value are treated as equal. Let p be the number of pixels per meter. Then two distances are allowed to differ up to a minimum of $2/p$ and 10% of the first distance. Also, points with a "smallest distance" to the pixel must have either x- or y-coordinates that differ by more than 0.2 m to be counted. This is consistent with resolution of cadastral data. Finally, the angle between the vectors from the pixel to these points have to enclose an angle greater than a threshold angle of 45°.

The medial axis formed from pixels is now transformed into a tree structure. Starting at an intersection point of a counter bearing edge and the modified medial axis, the tree is built iteratively. If counter bearings cannot be determined, the algorithm starts at an intersection of the footprint with the non-modified medial axis. The intersection point is the first node, i.e., the root of the tree. To add more nodes, all points of the medial axis are determined on a circle around the point of the current node. Only points on the circle with a mutual angle above 45° define nodes connected to the current node. Before dealing with these nodes, all points of the medial axis within the circle are removed. If there is no next node, then the algorithm searches for a nearest counter bearing point (or a nearest boundary point if there are no counter bearings) within the circle. If there exists one, a

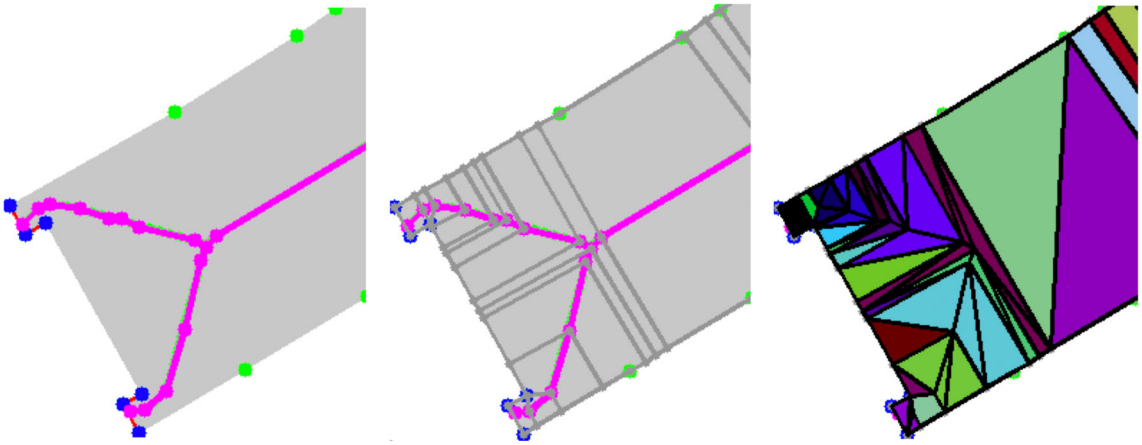

Fig. 9 Two counter bearings at one side of a bridge imply a bifurcated layout. The medial axis can be represented by a tree. The tree is used to divide the deck into smaller polygons (colored areas in the third image). If such a polygon is not planar, additional triangulation is required (black edges in the third image)

leaf of the tree is found. Otherwise the current branch does not lead to a counter bearing (or a boundary edge) and is discarded.

A z-coordinate (height value) is added to each node of the tree. Nodes of counter bearings inherit a z-coordinate from the neerest tree node. The coordinate of a tree node is calculated as the median of all z-coordinates of non-ground laser scanning points within a neighborhood with radius one meter. If no points can be found, the radius is iteratively increased to three meters. If the neighborhood still does not cover any non-ground points, the z-coordinates are similarly computed based on points representing ground.

Between the branch nodes (nodes with more than two adjacent edges), z-coordinates are smoothed. For modeling road surfaces, linear functions are fitted to sampled height values in Kada and Haala (2004) and linear, quadratic, or cubic functions are fitted to samples in Chen and Lo (2009). Here, a different approach is taken, which is adapted to typical height curves of bridges. This requires leaving z-coordinates of branch nodes untouched to fit with counter bearings and other segments of the bridge. Then, one interprets the z-coordinates as equidistantly sampled values of a continuously differentiable function $f : [0, \pi] \to \mathbb{R}$ so that

$$g(x) := \begin{cases} f(x) - f(0) - \frac{f(\pi)-f(0)}{\pi} x & \text{for } x \geq 0, \\ -f(-x) + f(0) - \frac{f(\pi)-f(0)}{\pi} x & \text{for } x < 0 \end{cases} \quad (1)$$

is an odd function on the interval $[-\pi, \pi]$ that can be continued with a period 2π. Then, g can be represented as a Fourier sine series:

$$g(x) = \sum_{k=1}^{\infty} b_k \sin(kx), \; b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(kx)\, dx.$$

Function g is smoothed by convolution with a Dirichlet kernel, i.e., a Fourier partial sum

$$\tilde{g}(x) = \sum_{k=1}^{n} b_k \sin(kx) \quad (2)$$

is a low-pass filtered version of g that satisfies $\tilde{g}(0) = \tilde{g}(\pi) = 0$. Based on experiments, $n = 5$ is chosen. The partial sum can be computed efficiently using the Fast Fourier Transform. Now the function f is replaced by

$$\tilde{f}(x) := \tilde{g}(x) + f(0) + \frac{f(\pi) - f(0)}{\pi} x$$

so that $\tilde{f}(0) = f(0)$ and $\tilde{f}(\pi) = f(\pi)$. Figure 4 shows two examples: The Fourier coefficient b_1 is dominant for both bridges, since the function g can be approximated by $\sin(x)$ quite well. This is true for most bridges with a non-constant slope. Therefore, the smoothing approach considering only low frequencies of a Fourier sine series seems suitable.

Unfortunately, there exist bridges that cross each other at different levels. Then z-coordinates of the highest bridge are wrongly assigned to lower bridges. Low-pass filtering

Fig. 10 Motorway bridge (Düsseldorf Airport Bridge) with a superstructure along the medial axis and multiple pillars from cadastral data

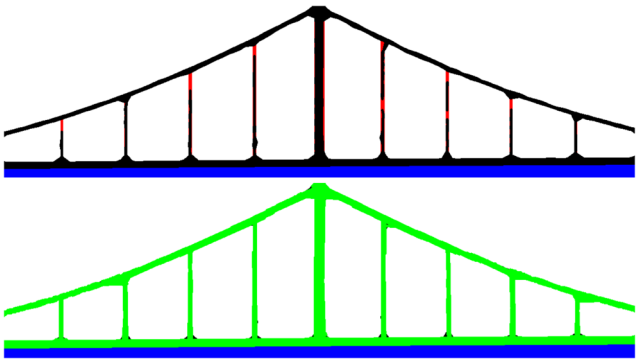

Fig. 11 Projected and connected points of a superstructure are shown in black. Red pixels were set using a column-based completion heuristics. Green pixels belong to line segments detected with a Hough Transform. Pixels below the deck are blue

is able to eliminate false height values on small intervals (see Fig. 5). In order to cope with larger disturbances, jump discontinuities are detected on segments between branch nodes. Only height differences above 3 m are considered. Then, z-coordinates between an upward and subsequent downward jump are replaced by linear interpolation. If only isolated coordinates deviate from the median of all z-coordinates of the segment, these are also replaced.

Then, the segments between branch nodes are simplified using the Douglas-Peucker algorithm to obtain a tree with few nodes and edges; see nodes and edges along the medial axis in Fig. 6. In the Douglas-Peucker algorithm, the distances of vertices to a straight line through the end vertices of a segment are calculated. If the largest distance exceeds a threshold value, then the segment is divided into two parts at the vertex with the largest distance. This vertex becomes part of the simplified tree, and the algorithm is recursively applied to the two new segments. Since the goal is a smooth surface, the simplification algorithm should be more sensitive to variations in the z-coordinates. Therefore, the z-coordinates are weighted by a factor of 5 before the distances are computed.

Skeleton-Based Graph

The next step is to create a graph from the simplified medial axis tree that contains important edges of the later deck polygons. To this end, the graph is initialized with the footprint polygon of the bridge. Corresponding edges are marked as outer edges. Then, the algorithm iterates through all nodes of the tree that are inside the footprint. For each of these nodes, new cross-connecting edges are inserted from the node to the two nearest footprint points outside the counter bearing lines if the new edges do not cross other edges. If a nearest footprint point does not coincide with a vertex of the graph, it is added and connected with

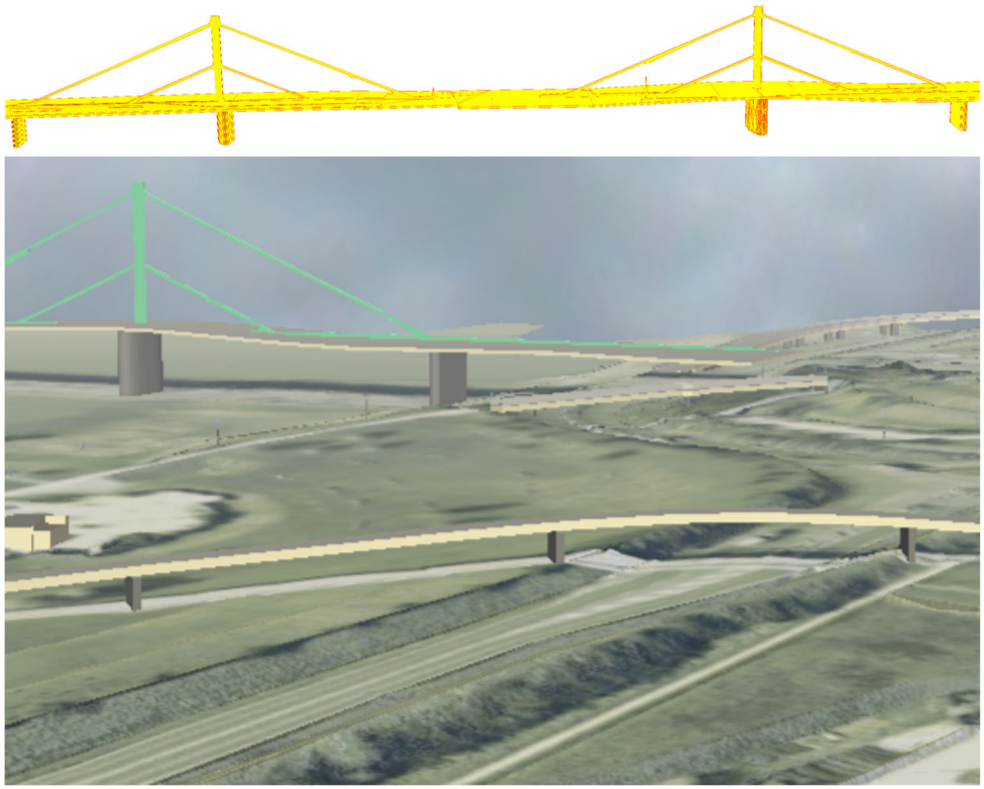

Fig. 12 The cable-stayed Leverkusen Bridge is given by two cadastral footprints because it connects two different cities. The superstructure of each part was reconstructed from points projected to only one plane. After connecting pixels, a Hough transform was used to find line segments. The reconstructed superstructure is based on the union of these line segments

Fig. 13 Superstructure reconstructed from point cloud: Three planes with more than 100 inlier points were detected above deck level of this segment of the Uerdingen Bridge. The cable structures and pylons on both sides of the bridge are each in a plane. The pedestrian walkways separate these planes from the cadastral footprint. The third plane defines a connection between the two pylons

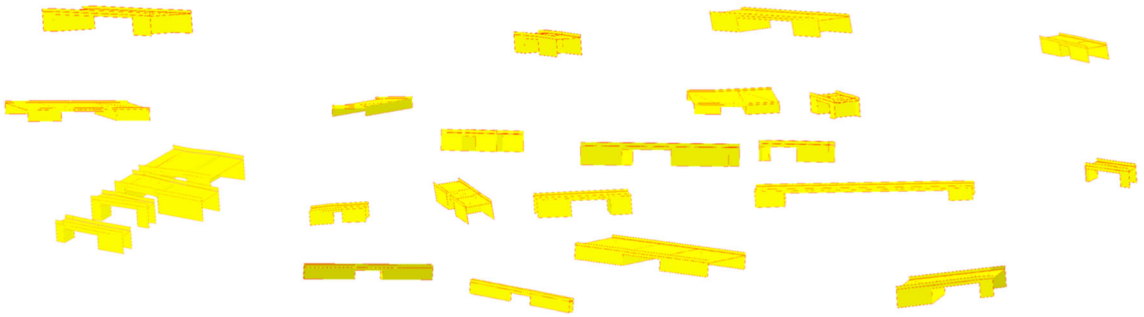

Fig. 14 Most bridges have a simple structure. In the examples, counter bearings are extended to fill gaps between floating parts and ground

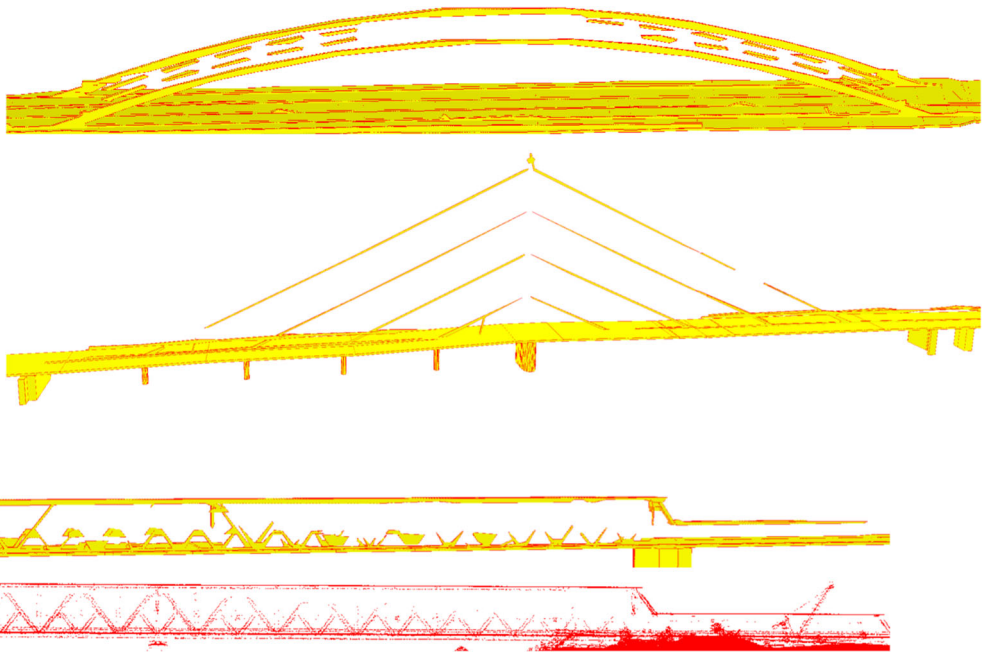

Fig. 15 The cables holding up the Bridge of Solidarity in the city of Duisburg (tied-arch bridge) are too small to be detected by airborne laser scanning. The pylon of Oberkassel Bridge in Düsseldorf is not visible in the laser scanning point cloud

Fig. 16 The point cloud (red) of the Rheinhausen railroad bridge is too sparse to reconstruct the steel truss completely (yellow)

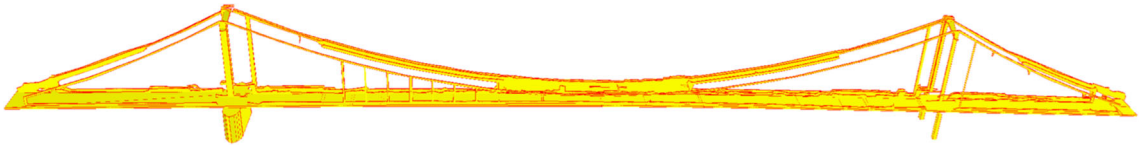

Fig. 17 The longest German suspension bridge is located in Emmerich. The point cloud was too sparse to find most of the vertical cables, and construction work disturbed the shape of the main cables

outer edges to the two adjacent vertices of the graph on the footprint polygon. The direct edge between the adjacent vertices is removed. If the two cross-connecting edges are on a straight line, they are merged and only one merged cross-connecting edge is added. Then the tree node is not needed in the graph. Otherwise, both edges and their tree node are added as a cross-connection. An example of a resulting graph is shown with gray vertices and edges in Fig. 6.

The z-coordinate of the tree node is copied to the two border nodes of the cross-connection. If a node is assigned more than one z-coordinate, the median is calculated. Cross-connections are often orthogonal to tree edges, i.e., they can be orthogonal to a path over the bridge. Their use as tesselation edges supports a smooth modeling of deck curvature.

Deck Polygons

Individual polygons are now iteratively derived from the graph (see Fig. 7). The first iteration starts at an arbitrary footprint node. Then, the graph is traversed in the mathematically positive sense along the outer edges until a node is reached that has at least one cross-connecting edge. Without having visited other nodes before, this node could directly be the start node. The node is stored as a future start node. Now, the next node is selected by following the leftmost cross-connection. After passing the cross-connection, the graph is traversed along outer edges until a next cross-connection is reached, which then is followed, and so on. This ends with a return to the start node. Then a polygon is derived through the visited nodes. Traversed cross-connecting edges become outer edges, and traversed outer edges are removed from the graph. Then the next iteration starts at a stored future start node.

After a polygon has been constructed, it is checked whether nodes with existing z-coordinates lie on a single plane. If this is the case, all nodes without a defined z-coordinate are assigned a corresponding coordinate of this plane. Otherwise, if there are more than three nodes, the polygon is divided into two or three smaller polygons by adding one or two additional edges. In most cases, exactly one cross-connection is traversed. Therefore, only the first visited cross-connection is considered. If it consists of one edge, the additional edge runs from the start node of the iteration to the end of the cross-connection (or, if there is no cross-connection, to the end of a counter bearing). This is the reason for the triangles in Fig. 7. If the cross-connection consists of two edges, then up to two additional edges are introduced. One runs from the start node to the middle node of the cross-connection. If nodes of a new polygon still do not define a single plane, a second edge is inserted that connects the node preceding the start node to the middle node (see Fig. 8). Additional edges are introduced only if they are completely inside the polygon.

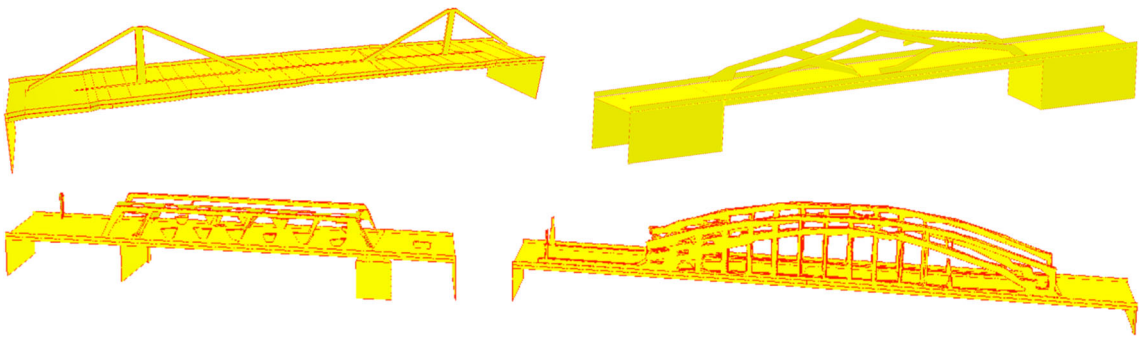

Fig. 18 Examples of smaller bridges with superstructures: Jülich Street Bridge in Düsseldorf, harbor bridge in Krefeld, and Karl Lehr Bridges over the harbor canal and the Ruhr in Duisburg

Table 1 Visual inspection of computed bridges

	Krefeld	Leverkusen
Correct	162	248
Inconsistent counter bearings	2	9
Inconsistent deck heights	30	24
Trees as superstructures	3	4

Now one has to deal with z-coordinates of the vertices of constructed polygons. If the vertices of a polygon that already have z-coordinates define a planar plane, then this plane defines all its missing z-coordinates. In rare cases, no plane can be determined that supports all nodes. In this case, the missing z-coordinates are determined using a plane that best supports the given 3D nodes. Then, a 3D triangulation of the polygon is performed (using an OpenGL library function) to obtain planar surfaces (see black lines in Fig. 9).

Approximation of Superstructures and CityGML Generation

Bridges may have superstructures above the deck level. The goal is to automatically provide models with recognizable superstructures. To this end, a modified version of RANSAC is applied to find planes that support many points above the deck. For example, the cables of suspension bridges are often located in one or two planes. Therefore, RANSAC is applied to find all planes with more than 100 inlier points (closer to the plane than a threshold value of 2 m) that are at least 2 m above the highest point of the deck. This distance from the deck is necessary to ignore noise, and the threshold value of 100 points is chosen for point clouds with four to ten points per square meter (cf. Section 1). Since we only consider points that are at least 2 m above the floating part of a deck, the corresponding point cloud is often of small size. However, we observed a maximum point count of nearly 100,000. Based on the examples discussed in Section 1, one can assume that a largest relevant plane has at least more than 15% inlier points. To find such a plane with 99.9% confidence, $\ln(1-0.999)/\ln\left(1-(0.15)^3\right) \approx 2043$ iterations are sufficient.

One difficulty of RANSAC is that planes also have many support points, even though they do not appear in the model but merely intersect many other, actually existing planes. When detecting roof surfaces in 3D city modeling, this is avoided by only considering those points as possible inliers whose point normals are approximately parallel to the plane normal. However, this does not fit the projection approach chosen here. Therefore, we take advantage of the fact that superstructures often run parallel or orthogonal to the edges of cadastral footprints. Therefore, if possible, planes are slightly adjusted (up to an angle of 5°) to intersect the ground plane in a line parallel or orthogonal to the longest footprint edge. Nearly vertical or horizontal planes are also replaced with exactly vertical or horizontal planes. This is all done by applying a RANSAC algorithm similar to Goebbels and Pohle-Fröhlich (2020). This algorithm also performs a principle component analysis to minimize squared distances between inlier points and the candidate plane (cf. Fischler and Bolles (1981)). Experiments have shown that vertical planes tend to describe superstructures better than horizontal planes. Therefore, vertical planes with an angle of inclination greater than 45° are first detected until the number of inliers has halved compared to the first detected plane. Only then are any planes considered.

For each identified plane, a 3D rectangle is calculated that lies on the plane and covers the projections of all inlier points. The rectangle is extended so that it reaches down to

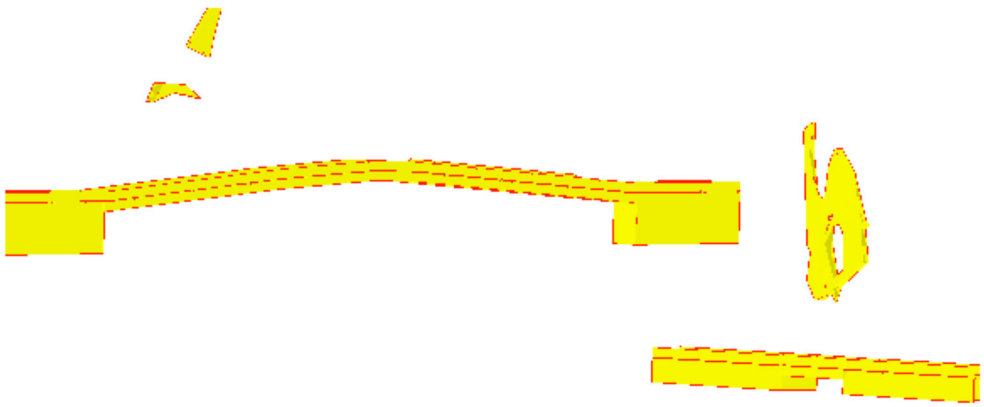

Fig. 19 Trees are misinterpreted as superstructures: They could be automatically removed if they are high above the deck

Fig. 20 The bridge is completely covered by trees. As a result, the z-coordinates of the deck are wrong

the lowest deck elevation. This is necessary to avoid gaps between deck and the detection space that starts 2 m above the highest deck point. Based on the rectangle, a binary raster image is generated by orthogonally projecting all laser scanning points onto the plane that are closer to the plane than 2 m, as before. But now also points below the highest point of the deck plus 2 m have to be considered. Since the deck may not be planar, the deck is projected onto the plane as well. Then all points below the deck curve are removed from the raster image. Each pixel represents an area of 10 cm $\times$ 10 cm. The resolutions of the image and point cloud are consistent. Due to sparsity, most pixels of the superstructure are isolated. Therefore, one needs to generate a coherent structure by connecting pixels. If a projected point has two or more neighbors within a distance of 2 m, its pixel is connected to its neighbors by line segments. The result is an image of the superstructure. However, its lines may not be connected. To connect line segments, two methods were tested (see Fig. 11):

- Maxima of a column histogram of the image are utilized. Each column with a local maximum of superstructure pixels probably represents a pylon or a cable of a suspension bridge. In such columns, the first and the last superstructure pixels are connected with a line.
- Line segments are detected if they are supported by at least 20 pixels. To this end, a Hough transform is applied by using OpenCV function HoughLinesP. In examples, some line segments showed gaps up to a length of 4 m. The parameters of HoughLinesP are chosen accordingly. Line segments are then drawn into the image to fill in gaps. For example, this function was applied to reconstruct the cable-stayed bridges in Figs. 10 and 12.

Both methods provide an image from which contours can be detected (Fig. 11).

Such contours can have openings. Contour polygons are simplified with the Douglas-Peucker algorithm. The outer contours and their openings then are used as superstructure polygons; e.g., see Figs. 10, 12, and 13. In most cases, it is sufficient to only consider the largest outer contour. By

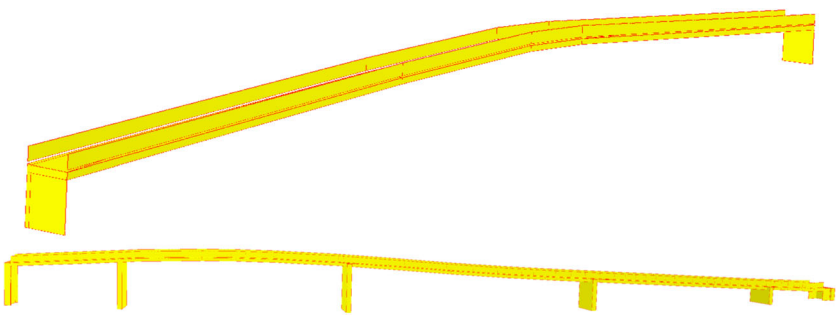

Fig. 21 Reconstructed pedestrian bridge 1, reconstructed interchange bridge

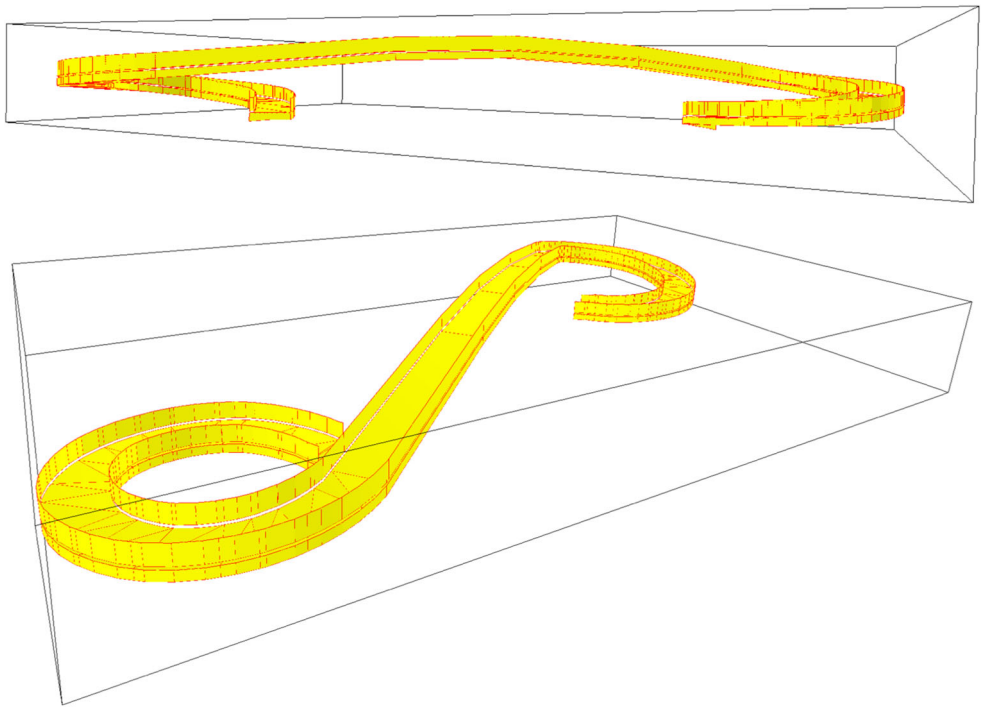

Fig. 22 Reconstructed pedestrian bridge 2

manually editing the image (which has not been done in this study), the quality of the models can be increased without using CAD tools. The polygons are extruded in the direction of the normal of their plane to become 3D solids.

Finally, a CityGML structure is created. To this end, the deck is extruded to become a 3D object as well by writing the CityGML objects OuterFloorSurface, OuterCeilingSurface, and WallSurface. Counter bearing walls and pillar walls are also added based on their cadastral footprints (see, e.g., Figs. 10, 12, 13, and 21). These walls extend from ground (as determined from point clouds) to the deck. Counter bearings and pillars are represented by BridgeConstructionElement objects. Counter bearings can be extruded to fill gaps between the floating part of the bridge and the terrain model (cf. Fig. 14). Such extensions and superstructure polygons are also modeled separately as BridgeConstructionElement objects. In addition, railings are added generically as elements of type OuterBridgeInstallation. Depending on the application scenario, one can choose to use counter bearing objects and

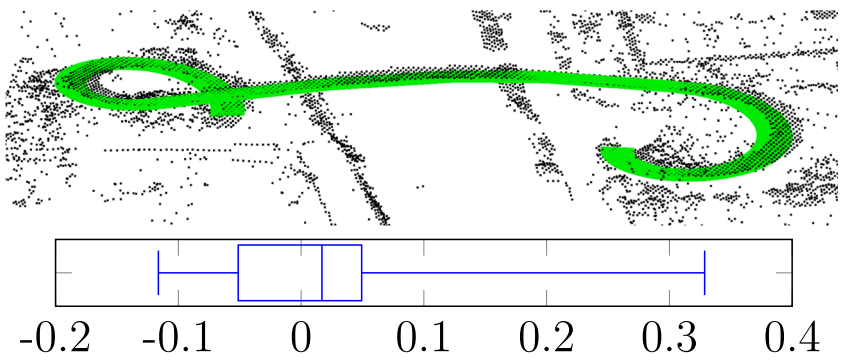

Fig. 23 Comparison between reconstructed deck of pedestrian bridge 2 and point cloud; the box plot shows the distribution of z-coordinates of the reconstructed deck minus z-coordinates of points along the medial axis. The differences are measured in meters

Fig. 24 Roofs lead to wrong z-coordinates of decks of pedestrian bridges at a railroad station

their extensions to connect floating parts with ground. When modeling crossings in road networks, only the floating part is relevant.

Results

The experiments were based on airborne laser scanning point clouds that are available from Geobasis NRW.[1] These point clouds have a resolution between 4 and 10 points per square meter which varies from city to city. Only "last return" points were considered. Full wave laser scanning was not available. Points were classified. We use the distinction between points belonging to ground and points belonging to points above ground. Comparison with cadastral data of buildings shows that the points deviate significantly less than 10 cm from their actual position. It should be noted, however, that some cadastral footprints are also inaccurate (see Fig. 23).

The algorithm was applied to 482 bridges in the North Rhine-Westphalian cities of Leverkusen and Krefeld. The majority of these bridges has a simple layout (see Fig. 14). But the data also include two bridges spanning the Rhine River.

These are the only bridges with significant superstructures. To further test the reconstruction of such structures, other large bridges built over the Rhine River were also considered (see Figs. 10, 15, 16, 17, 25, and 26).

Some smaller bridges were also selected for exemplary tests (see Fig. 18).

Table 1 shows results for Krefeld and Leverkusen. In total, 85% of bridges have a correct appearance. There are some bridges with cadastral counter bearing information which is not complete and could not be completed automatically. A larger number of brides show inconsistent deck elevations. Reasons are vegetation and roof structures. In some cases, trees were interpreted as superstructures (see Fig. 19). Trees were also a problem for a few small bridges in forest areas (see Fig. 20).

Especially for bridges with non-constant slopes like pedestrian bridges, the method generated smooth deck surfaces (see Figs. 21 and 22).

All bridges of each square kilometer tile were calculated together. After loading the point clouds of the tile and its neighboring tiles, the calculation per bridge took at most a few seconds on a laptop with i5 processor.

Figure 23 shows the reconstructed deck of the bridge in Fig. 22 together with the underlying point cloud. Points above the deck are visible; points below the surface are hidden. It can be seen that smoothing leads to small deviations. Furthermore, the cadastral footprint does not exactly match the point cloud. This leads to a small shift that fortunately does not really change the elevation values along the medial axis. Such errors are common when using data from different sources.

Some bridges have superstructures along the medial axis. In most cases, the calculation of the median was sufficient to ignore superstructure points when computing z-coordinates of the decks. However, using the median did not work for the bridge in Fig. 25. Here, the 0.25 quartile provided correct deck information. However, a general replacement of the median by the 0.25 quartile led to problems with bridges with inaccurate footprints and adjacent vegetation.

[1] https://www.bezreg-koeln.nrw.de/brk_internet/geobasis/hoehenmodelle/3d-messdaten/index.html

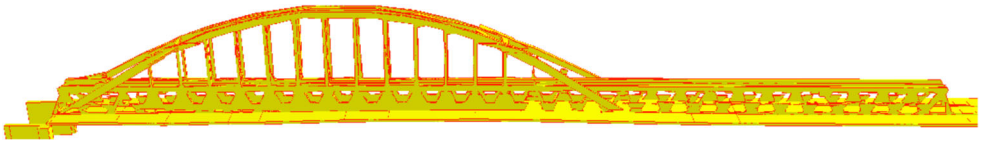

Fig. 25 The Düsseldorf-Hamm railroad bridge is a combined steel truss and tied-arch bridge. Most laser scanning points belong to the massive steel truss, so the computation of the deck heights based on the median instead of the 0.25 quartile of z-coordinates does not work properly

Conclusion

The significance of the study lies in the fact that all bridges of large land areas can be reconstructed from existing cadastral footprints and airborne laser scanning data without manual interaction and without major hardware requirements. Models comply with CityGML level of detail 2. Depending on the point cloud density, even pylons and cables from suspension bridges can be added automatically, in some cases achieving a higher level of detail.

However, there are limitations that require further work.

- If the deck of a bridge is inconveniently occluded at branch nodes of the medial axis tree by other building parts such as roofs (see Fig. 24) or other bridges, this cannot be corrected by the presented version of the algorithm.
- The medial axis must induce a tree unlike a general graph with cycles. Such a graph can occur with a traffic circle as a bridge deck. To deal with such bridges, the tree generation algorithm has to be extended to take into account the cycles.
- The method only considers slope in longitudinal direction, i.e., along the medial axis. There also might be slope in latitudinal direction.
- Railings are added to non-counter bearing edges in a model-based fashion. A comparison of deck heights with laser scanning points did not yield appropriate data-based heuristics for deciding where to place railings.
- Point clouds from airborne laser scanning are not suitable for reconstructing structures below the deck because they are at least partially occluded.

Fig. 26 Large bridges in Cologne: superstructures were reconstructed using the Hough transform combined with histogram-based column completion on projection images. Yellow bridges from top to bottom and left to right: Hohenzollern Bridge, South Bridge, Mülheim Bridge (based on two cadastral footprints), two images of Rodenkirchen Bridge. Hohenzollern Bridge consists of 3 × 3 bridge sections, each with two steel arches on its sides. The arches can be captured with four vertical parallel planes. However, horizontal planes that intersect the arches also have a large number of inliers. Therefore, it was necessary to detect vertical planes prior to horizontal planes

Acknowledgements I would like to thank Johannes Scharmann from the cadastral office of the City of Leverkusen for his support with data, expertise, and images of the computed bridges which have been imported into the Leverkusen 3D city model. I also thank my colleague Regina Pohle-Fröhlich for joint work on building reconstruction that I could apply in this study.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of Interest The author declares no competing interests.

References

Biljecki F, Stoter J, Ledoux H, Zlatanova S, Çöltekin A (2015) Applications of 3D city models: State of the art review. ISPRS Int J Geo-Inf 4:2842–2889

Chen LC, Lo CY (2009) 3D road modeling via the integration of large-scale topomaps and airborne LIDAR data. J Chin Inst Eng 32(6):811–823

Cheng L, Wu Y, Wang Y, Zhong L, Chen Y, Li M (2015) Three-dimensional reconstruction of large multilayer interchange bridge using airborne LiDAR data. IEEE J Sel Top Appl Earth Obs Remote Sens 8(2):691–708

Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395

Goebbels S, Pohle-Fröhlich R (2016) Roof reconstruction from airborne laser scanning data based on image processing methods. ISPRS Ann Photogramm Remote Sens and Spatial Inf Sci III-3:407–414

Goebbels S, Pohle-Fröhlich R (2020) RANSAC for aligned planes with application to roof plane detection in point clouds. In: Proceedings of international conference on computer graphics theory and applications (GRAPP), pp 193–200

Gröger G, Kolbe TH, Nagel C, Häfele KH (2012) OpenGIS city geography markup language (CityGML) encoding standard. Version 2.0.0. Open Geospatial Consortium

He Y (2015) Automated 3D building modeling from airborne LiDAR Data. Dissertation, University of Melbourne

Henn A, Gröger G, Stroh V, Plümer L (2013) Model driven reconstruction of roofs from sparse LIDAR point clouds. ISPRS J Photogramm Remote Sens 76:17–29

Hu F, Zhao J, Huang Y, Li H (2020) Learning structural graph layouts and 3D shapes for long span bridges 3D reconstruction. arXiv:1907.03387v2:1–27

Kada M, Haala N (2004) Integration of street networks and LIDAR for modeling and visualization of terrain data. In: Proceedings of Asian conference on remote sensing 25th, Chiang Mai, Thailand, pp 256–261

Kutzner T, Chaturvedi K, Kolbe TH (2020) CityGML 3.0: New functions open up new applications. PFG 88:43–61

Perera SN, Maas NG (2012) A topology based approach for the generation and regularization of roof outlines in airborne laser scanning data. In: Seyfert E (ed) DGPF Tagungsband 21, DGPF, Potsdam, pp 400–409

Sithole G, Vosselman G (2003) Automatic structure detection in a point cloud of an urban landscape. In: Proceedings 2nd GRSS/ISPRS joint workshop on remote sensing and data fusion over urban areas, URBAN2003, pp 67–71

Sithole G, Vosselman G (2006) Bridge detection in airborne laser scanner data. ISPRS J Photogramm Remote Sens 61(1):33–46

Wang R, Peethambaran J, Chen D (2018) LiDAR point clouds to 3-D urban models: a review. IEEE J Sel Top Appl Earth Obs Remote Sens 11(2):606–627