Research Article

Automatic generation of cross sections from computed tomography data of mechanical joining elements for quality analysis

T. M. Schromm[1] · C. U. Grosse[1]

Received: 16 January 2021 / Accepted: 15 September 2021
Published online: 09 October 2021

Abstract
In this work, we present a methodology for shifting from a conventionally destructive, manual quality analysis for repetitive processes towards a non-destructive and largely automated process. The objects subjected to the quality analysis are mechanical joining elements like rivets or flow-drilling screws. We propose an algorithm that can automatically find and extract such joining elements from a computed tomography (CT) scan, rotate these elements to an upright orientation and eventually generate radial cross sections parallel to the elements' longitudinal axis. The proposed algorithm was tested on five grayscale-based computed tomography volumes, with one synthetically generated volume. We will discuss both, cases in which the duo of CT and our proposed algorithm produces satisfying results, as well as cases in which it fails. Limitations of both the scan acquisition process and the proposed algorithm will be elaborated on and potential improvements will be mentioned.

Keywords Non-destructive testing · Computed tomography · Joining technology · Automation · Image processing

1 Introduction

The quality of mechanical joints, like rivets or screws, is conventionally analyzed with destructive methods, like macro-sectioning (Fig. 1b) or shear and fatigue tests [1–4]. Macro-sectioning provides structural 2D information and is usually performed through the structure's center and parallel to its longitudinal axis [5]. Shear or fatigue tests provide information regarding the joint's strength. While these methods have proven to be very effective, they are exclusive, meaning that conducting one makes conducting the other impossible. The inherent lack of volumetric information and the destructive nature make current joint inspection therefore both inefficient and unsustainable.

Industrial computed tomography (CT) has the potential to provide a detailed[1] virtual volumetric representation of even large and highly attenuating objects like cars or shipping containers [6]. Therefore, by introducing industrial computed tomography into the inspection process of joining elements, it is possible to perform destructive testing (shear, fatigue) on a sample whose detailed, structural information has already been acquired, cutting the number of samples needed for the gained information in half. In short, this means that the process sustainability was doubled while, at the same time, the amount of accessible information was shifted from 2D to 3D. While CT seems to be a promising solution to the stated lack of information and sustainability, it comes at a cost. On one

[1] In the industrial context, nano-computed tomography systems are able to resolve features down to several hundred nanometers [6]. The case-specific resolution depends on the constellation of x-ray source, detector and sample, as well as the expertise of the user.

✉ T. M. Schromm, thomas.schromm@tum.de; C. U. Grosse, grosse@tum.de | [1]Chair of Non-Destructive Testing, Technical University of Munich, Franz Langinger Str. 10, 81245 Munich, Germany.

hand, more accessible information means more evaluation is required. Evaluation often involves repetitive workflows with many manual operations. The exact number of operations depends on the specific case. In the case of joining elements, an exemplary workflow can include the following: After loading the volume in a suitable software, the user would adapt the grayscale values until the contrast between air, joined structures and joining elements is optimal for human vision. Next, the elements would need to be aligned and a cross-sectional plane would need to be placed through its center and parallel to its longitudinal axis. Once relevant features have been dimensioned, the radial cross sections need to be exported for a report. Such a workflow can be especially tedious when it has to be performed for a multitude of elements that are originally aligned differently. On the other hand, computed tomography is susceptible to artifacts (see for example [7, 8]) and struggles with resolving neighbouring structures that do not differ sufficiently in density, geometry or elementary composition [9]. This inherent problem is illustrated in Fig. 1c and d.

This work investigates the possibility and applicability of an algorithm that automates the previously mentioned workflow, except the dimensioning part. The proposed algorithm can automatically find and extract characteristic elements, like joining elements, from a volumetric CT scan, rotate these elements to an upright orientation and finally generate radial cross sections parallel to the elements' longitudinal axis. The cases in which the algorithm is not or only partially applicable will also be discussed. This work does not deal with selecting optimal scan parameters, CT systems or with preparing the samples in order to optimize the scan quality. Only qualitative comments will be made at the appropriate passages in the text. Special artifact reduction algorithms were neither applied nor investigated. There was also no investigation to find an optimal trade-off between resolution and number of joining elements per volume. Nevertheless, the algorithm's functionality and its potential to increase evaluation efficiency for certain cases will be presented and discussed. Furthermore, we intend to exploit this algorithm's functionality to quickly produce unlabeled training data for various machine learning (ML) applications like pose estimation, joint dimensioning and semantic segmentation. If such ML algorithms are implemented, this would complete the automated quality analysis process for mechanical joining elements. However, this work does not deal with machine learning algorithms beyond mentioning them as a complementary means for a fully non-destructive and automated quality analysis.

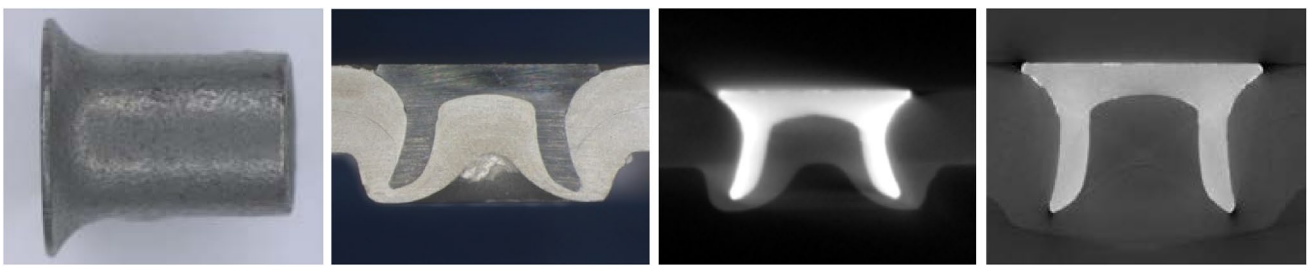
2 Methods

In this section, the samples and the resulting reconstructed CT volumes that were used to verify the proper functioning of the algorithm are described. Important scan parameters and the CT system used for every volume will be listed. Four conditions are imposed on the samples, which must be met for the proposed algorithm to function reliably.

2.1 Sample description and scans

Three types of joining components have been used for this, namely *flow-drilling screws* (FDSs), *self-piercing rivets* (SPRs) with a half-hollow body and *flat head rivets* (FHRs). All three are widespread methods of joining structures in the automotive and aeronautics industry (see for example [6, 10–13]).

The algorithm's performance was tested on five volumes in total - four physical volumes (Vol1-Vol4 in Fig. 2) and one synthetic volume (Vol5 in Fig. 3), which consists of two randomly aligned joints from each physical volume. The motivation for this artificial volume is to investigate the algorithm's performance in the presence of different



(a) Unprocessed rivet (b) Macro section (c) CT with strong artifacts (d) CT with few artifacts

Fig. 1 **a** Close-up of an unprocessed rivet. **b–d** Three different rivets from three different acquisition procedures. The contrast between background and joined aluminum plates is rather weak in both CT cross sections

types of joining elements. Important scan parameters that were used to acquire the volumes and additional information are listed in Table 1. Among other information, the acceleration voltage U, the heating current of the cathode I, the focal spot to detector distance FDD, the focal spot to object distance FOD and the geometric magnification M.

In Vol1, the four FDSs are made of the steel alloy *23MnB4*. The top and bottom plates measure 2 mm and 3 mm in thickness, respectively. Vol2 contains four rivets made of Boron steel *38B2*, which were intentionally damaged before processing. Each rivet shows a previously induced cut that was forced to stretch during the riveting process. Both the bottom and top plates measure 1.5 mm in thickness. Vol3 contains eight rivets in total, all made of Boron steel *38B2*. Both top plates measure 2.5 mm and 1.5 mm in thickness. All plates in Vol1 to Vol3 are made of aluminum.

The brake disk in Vol4 was mostly removed in order to minimize its effect on the rivets during the CT scan. The brake is made of cast iron *GS 97075*. The brake and the aluminum pot are joined by eighteen flat head rivets, which are made of the steel alloy *X3CrNiCu18-9-4*.

2.2 Conditions imposed on the samples

There are four conditions that must be met by a sample in general for the algorithm to function properly:

(1) Difference in contrast The grayscale values of the joining elements must be sufficiently different with regard to the surrounding joined structures. This translates to elements having sufficiently different resolvable dimensions, density or atomic number. Otherwise, manual interaction is needed in order to accurately isolate only the joining elements. This, however, would defeat the purpose of the algorithm and nullify its advantage over a standard manual approach.

(2) Difference in number of voxels If additional structures with similar grayscale values are present in the volume, the number of voxels of which they are comprised must differ significantly from the number of voxels that comprise the joining elements the algorithm is supposed to find. The difference must be large enough that a thresholding operation (for example median or mean) applied to the individual amount of element-voxels will be able to clearly differentiate between relevant and irrelevant structures.

(3) Spatial separation of elements Joining elements must not touch each other. Otherwise the algorithm is going to treat them as a single structure. Due to limited resolution and/or artifacts this can also hap-

Table 1 Scan parameters and additional information relevant for acquiring the volumes on which the algorithm was tested. The number of voxels refers to the reconstructed volume

	Scan parameters							System specifications		Sample	
	U	I	FDD	FOD	M	Filter	Voxelsize	Detector	System	Materials	#Joints
	[kV]	[mA]	[mm]	[mm]	[-]	[mm]	[μm]	[-]	[-]	[-]	[-]
Vol1	220	140	815.84	117.19	4.60	Cu: 0.5 Sn: 0.5	43.4	GE DXR250	v|tome|x M 240D	FDS: Steel Plates: Al	
Vol2	270	310	690.25	205.11	3.37	Cu: 1.0	59.4	GE dynamic41|100	v|tome|x L 300D	SPR: Steel Plates: Al	4×SPRs
Vol3	240	350	1100.00	315.52	3.49	Cu: 1.0	28.7	GE dynamic41|100	v|tome|x L 300D	SPR: Steel Plates: Al	8×SPRs
Vol4*	270	300	942.90	536.63	1.76	Cu. 1.0	113.8	GE dynamic41|100	v|tome|x L 300D	Disk: Cast iron Pot: Al FHR: Steel	18×FHRs
Vol5	*Compound volume: Contains 2×FDS subvolumes from Vol1, 2×SPR subvolumes from Vol2, and 4×SPR subvolumes from Vol3										

*To reduce artifacts, a helix scan was performed with five calibration images, a total rotation angle of 357.52° and a total feed range of 226.09 mm

pen if the elements do not have physical contact, yet are processed too close together.

(4) Geometric shape of element The joining element's covariance matrix (spatial abstraction) must provide at least one eigenvalue that is either sufficiently larger or smaller than the other two eigenvalues. Otherwise it is not clear which axis to align with the reference. Ideally, the joining element's eigenvectors coincide with its axis of symmetry, provided (pseudo-) symmetry exists at least around one axis.

These conditions limit the algorithm's usability in the family of joining technologies by dismissing, for example, clinching joints (no joining element to detect), welds (not enough difference in grayscale values compared to the surrounding material), joints incorporating both steel structures and steel joining components (joined structures and joining elements not separable based only on grayscale values), adhesives (barely to not at all visible when used to join metal parts).

Nevertheless, the algorithm contributes significantly to increasing process efficiency in certain cases, as will be shown in sects. 3 and 4.

2.3 Proposed algorithm

The proposed algorithm is designed to work on grayscale value-based, volumetric CT data. Its abstracted functionality can be described as follows: It finds and rotates elements, like the ones described in subsect. 2.1, to a preferred orientation and generates radial cross sections along the elements' longitudinal axis. The algorithm was written with the intention of requiring as little user interaction as possible. In the cases of Vol1-Vol3 and Vol5, the algorithm only needed the respective path to the required files as input. In the case of Vol4, additional user action was required. The respective cases and performance will be presented and discussed in sects. 3 and 4. The algorithm consists of three main parts, which will be elaborated below.

2.3.1 Data import

First, the algorithm imports a *.vol* file and the respective *.pcr* file. Both files are automatically created by the reconstruction software (GE[2] Phoenix datos|x) of the CT system (GE v|tome|x L/M) used. The *.vol* file contains the reconstructed volume (voxels with grayscale value) of the CT scan. The *.pcr* file contains information about the scan parameters and the dimensions of the reconstructed volume. Both files are necessary in order to properly load and assemble the reconstructed volume into MATLAB. However the workflow and functionality can, in principle, also be translated to other programming languages due to the unrestricted readability of the aforementioned GE scan files (*.vol* and *.pcr*). Importing equivalent system files of other industrial CT manufacturers were not investigated. Therefore, we cannot confidently generalize the applicability of our algorithm to environments with CT systems by other brands. Let us assume, that there are more than one brand of CT systems, whose scan files are utilizable. To keep the required user interactions to a minimum, incorporating other brand's scan files could be tackled with either a brand type query for the user or an automated brand file detection functionality in the beginning of the algorithm.

A simple alternative to reading in the *.vol* file, that would work across brands, is the sequential import of image files that, put together, make up the volume. However, this requires a preceding software that is capable of reading in a *.vol* file and exporting it as a set of image slices. This approach was successfully tested in the beginning with VolumeGraphics MAX 3.4. However, for reasons of efficiency, reading in the *.vol* file was implemented in the final version of the presented algorithm.

A script was written (see Fig. 4), which can successively load the bytes from the *.vol* file in smaller packages and rebuild the reconstructed volume package by package. This way, live status updates can be printed out, which gives the user a rough estimate of the remaining time needed to import the volume. The following variables are required for this process: The total number of bytes b_{tot}, the bytes per voxel b_{vox} (mostly 2 bytes or, respectively, 16 bits per voxel in the case of CT), the volume dimensions (x_{vol}, y_{vol}, z_{vol}), the number of volume slices per regular package $N_{s,reg}$, the number of volume slices per residual package $N_{s,res}$, the maximum byte size of one package p_{max} and the number of packages N_p. A regular package is defined by the maximum byte size of one package. The difference between the complete volume and the collective size of regular packages defines the size of a residual package. The following formulas describe to the package-related arithmetic: Here, $\lceil ... \rceil$

$$b_{vox} = \frac{b_{tot}}{x_{vol} \cdot y_{vol} \cdot z_{vol}} \quad (1)$$

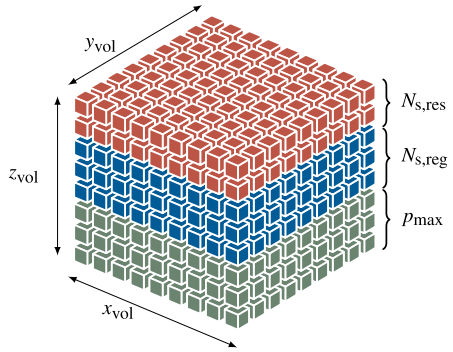
$$N_{s,\,reg} = \left\lfloor \frac{p\,max}{x_{vol} \cdot y_{vol} \cdot b_{vox}} \right\rfloor \quad (2)$$

[2] The branch of General Electric (GE), which develops and conceptualizes computed tomography systems, is now part of Waygate Technologies, which in turn belongs to Baker Hughes. Therefore the given product names might not be up to date.

$$N_{\mathrm{p}} = \left\lceil \frac{x_{\mathrm{Vol}}}{N_{\mathrm{s,\,reg}}} \right\rceil \tag{3}$$

$$N_{\mathrm{s,\,reg}} = \mathrm{mod}\,(z_{\mathrm{vol}},\, N_{\mathrm{s,\,reg}}) \tag{4}$$

and $\lfloor ... \rfloor$ are the ceiling and floor operator, respectively. By means of this arithmetic and MATLAB's predefined functions *fseek(...)* [15] and *fread(...)* [16], the scan data can be imported to MATLAB directly. A code excerpt of this can be seen in Fig. 4.



In a subsequent step, the algorithm reduces the volume's dimensions via binning, if requested by the user or required due to computational limitations. This depends on the available memory and the desired resolution in the final cross sections. This step, however, was only applied to Vol3, which otherwise could not have been handled by MATLAB. Since CT scans are comparatively large files, ranging from a few to several hundred gigabytes [6], it is advisable to consider the usable memory available on the computer. Otherwise MATLAB might run out of memory and throw an error. With the continuing trend of developing even higher resolving detectors [17], memory allocation for data processing will remain an important limiting factor.

The functionality from the code excerpt in Fig. 4 can be verbalized as follows: A scalar mapping function $f : V \rightarrow G$ maps every triplet-combination (index) of (i, j, k) from V to a positive rational number (intensity value) from G, with

$$V = \{(i,j,k) \,|\, i,j,k \in \mathbb{N} \wedge i \in [1, x_{\mathrm{vol}}], j \in [1, y_{\mathrm{vol}}], k \in [1, z_{\mathrm{vol}}]\} \tag{5}$$

and

$$G = \{g | g \in \mathbb{Q}^{+}\}. \tag{6}$$

While every triplet of V must be assigned to a number, not every number in G must be assigned by a triplet. This characteristic makes g a non-surjective, non-injective function.

2.3.2 Detection and isolation of relevant features

All n true features of interest that are contained in the volume V need to be extracted. In order for the algorithm to detect a true feature F_{ℓ}, every feature must be discernible from its virtual surroundings, based on its intensity value. Here, ℓ is the index of the respective feature with $\ell = 1...n$. In order to achieve this, Otsu thresholding [18] was applied. This thresholding approach automatically determines one or more thresholds t_{otsu} based on the histogram of the volume's intensity distribution by minimizing the intra-class intensity variance. MATLAB's *multithresh(...)* function [19], which does exactly that, was therefore applied to f, creating its binary equivalent $\hat{f}$. The thresholding operation can be described in simple mathematical terms as

$$\hat{f}(i,j,k) = \begin{cases} 1 \;\; if \quad f(i,j,k) > t_{\mathrm{otsu}} \\ 0 \;\; otherwise, \end{cases} \tag{7}$$

This strategy could also be applied to extract voxels of a certain grayscale value range. However, since the elements of interest were always comprised of the brightest voxels in the volume, i.e. the ones with the strongest attenuation, it was not implemented here. The binary features $\hat{F}_{\ell}$ are identified as independent, connected components and are extracted using MATLAB's *bwconncomp(...)* function [20]. A previously defined thresholding value helps to differentiate between true features of interest and noise or structures that are comprised of a lot fewer or a lot more voxels. The tolerance for Vol1-Vol3 and Vol5 is rather small (threshold equals 0.9 times the median voxel count of each feature), since all features of interest are approximately the same size. For Vol4, the tolerance is larger (threshold equals 0.2 times the median voxel count of each feature), since it contains features with very different sizes.

The remaining n individual binary features $\hat{F}_{\ell}$ are passed to the next part of the algorithm, where they are used to extract the corresponding grayscale features F_{ℓ} from the original distribution f. The extraction can be performed via

$$F_{\ell} = \{(i,j,k) \,|\, (i,j,k) \in \hat{F}_{\ell}\} \tag{8}$$

since the coordinates of $\hat{F}_{\ell}$ are absolute coordinates. A previously defined margin width $m \in \mathbb{N}_0$ was added to both binary and grayscale feature dimensions of $\hat{F}_{\ell}$ and F_{ℓ}, respectively. For this, minimal indices and maximal indices for the variables i, j, k are determined, via

$$
\begin{aligned}
i_{\min} &= \{i \mid \widehat{F}_\ell(i,j,k) = 1 \wedge min(i)\} and i_{\max} = \{i \mid \widehat{F}_\ell(i,j,k) = 1 \wedge max(i)\},\\
j_{\min} &= \{j \mid \widehat{F}_\ell(i,j,k) = 1 \wedge min(j)\} and j_{\max} = \{j \mid \widehat{F}_\ell(i,j,k) = 1 \wedge max(j)\},\\
k_{\min} &= \{k \mid \widehat{F}_\ell(i,j,k) = 1 \wedge min(k)\} and k_{\max} = \{k \mid \widehat{F}_\ell(i,j,k) = 1 \wedge max(k)\}.
\end{aligned} \tag{9}
$$

The resulting bounding box is then increased equally, perpendicular to all faces, by subtracting the margin from or adding it to the minimum or maximum index, respectively. For *i* this means

$$i_{\min} := i_{\min} - m \qquad and \qquad i_{\max} := i_{\max} + m. \tag{10}$$

Accordingly, similar adjusted statements hold for *j* and *k*. The additional margin provides more necessary information from a feature's virtual surroundings. Contextually, this would be more material of the joined structures that surround the joints. After this step, only the binary subvolumes and their grayscale counterparts remain in the memory. The imported volume (mappings *f* and $\widehat{f}$ and the set *V*) can be deleted to free up memory.

2.3.3 Evaluation of the feature orientation via principle component analysis

Now, every feature is represented by a binary and grayscale set of voxels. In the following, the binary set of voxels is used to determine an affine rotation matrix, which in turn is utilized to align the grayscale voxel set. After the alignment, the cross sections are generated and exported (see subsubsect. 2.3.4). The affine rotation matrix is determined using the *Rodrigues* rotation formula [21]. For this, a rotation angle and a rotational axis are required.

With MATLAB's *cov(...)* function [22] the covariance matrix $C_\ell = cov(\widehat{F}_\ell)$ is calculated. The next step is a principle component analysis (PCA) of every C_ℓ. The MATLAB function *eig(...)* [23] is applied to each C_ℓ to retrieve three eigenvectors $\boldsymbol{e}_{\ell q}$ of unit length and the three eigenvalues $\lambda_{\ell q}$, with $q = [1, 2, 3]$. These vectors are then used to determine the relative misalignment to the reference position, which is also represented by three vectors. The eigenvalues are used to scale the eigenvectors in order to differentiate them from each other based on their length: $\boldsymbol{v}_{\ell q} = \lambda_{\ell q} \cdot \boldsymbol{e}_{\ell q}$. The *scaled* eigenvectors $\boldsymbol{v}_{\ell q}$ of every C_ℓ indicate the spatial distribution of each $\widehat{F}_\ell$ as can be seen in the left part of Fig. 5. The eigenvectors $\boldsymbol{v}_{\ell q}$ can now be paired with their corresponding reference eigenvectors $\boldsymbol{v}_{\text{ref}, q}$ based on each vector's length. For this, three previously defined reference eigenvectors were included in the algorithm. In this work, only rotationally symmetric (rivets) or rotationally pseudosymmetric[3] (flow-drilling screws) samples are investigated.

The aim of the algorithm is to automatically rotate a detected element to a desired orientation. The contextual samples have one particular eigenvector (along their rotational symmetry axis), which can ideally be differentiated easily from the other two eigenvectors due to the fact that it is clearly longer (or shorter, as is the case for some rivets). This, combined with the samples' (pseudo-) symmetry makes it sufficient to align only this eigenvector with the corresponding reference eigenvector. This is illustrated in Fig. 5, where the only eigenvectors that need to be aligned with each other are $\boldsymbol{v}_{\ell 1}$ and $\boldsymbol{v}_{\text{ref},1}$.

Mathematically, the alignment can be performed with an *affine transformation matrix* in three dimensions

$$\boldsymbol{T} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & 0 \\ R_{21} & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{11}$$

For a rotation about an arbitrary axis, $\boldsymbol{R} = \{R_{ab} \mid a, b \in [1, 2, 3]\}$ can be calculated with the *Rodrigues* rotation formula:

$$\boldsymbol{R} = \boldsymbol{I} + (\sin\theta)\boldsymbol{K} + (1 - \cos\theta)\boldsymbol{K}^2, \tag{12}$$

with **I** being the 3×3 identity matrix. The matrix **K** represents the *cross-product matrix*

$$\boldsymbol{K} = \begin{bmatrix} 0 & -k_z & k_y \\ k_z & 0 & -k_x \\ -k_y & k_x & 0 \end{bmatrix}, \tag{13}$$

with $\boldsymbol{k} = \left[k_x, k_y, k_z\right]^T$ being the axis of rotation and θ the angle of rotation (in the example in Fig. 5, $\theta \equiv \alpha$). If there is a simultaneous translation, the last row and column in Equation 11 are nonzero. The axis of rotation **k** is the result of $\mathbf{k} = \boldsymbol{e}_{\ell q} \times \boldsymbol{e}_{\text{ref}, q}$, the cross-product of the eigenvector with the respective reference vector (see Fig. 5). Once the

[3] Contrary to rivets, flow-drilling screws do not possess a rotational symmetry axis, due to their asymmetric thread and head. However, the symmetry breaking voxels are few compared to the symmetry supporting voxels. Their influence on the covariance matrix and therefore the assignment of the eigenvalues $\lambda_{\ell q}$ to the respective eigenvectors $\boldsymbol{v}_{\text{ref}, q}$ was minimal, as can be seen in the results in sect. 3.

rotation matrix has been determined, the rotation of both the binary and grayscale subvolumes $\widehat{F}_\ell$ and F_ℓ, respectively, can be performed.

In the following, the subvolumes before rotation are referred to as *source*, and after rotation as *target*. Instead of using *forward mapping* (rotation of voxel from source to target), *inverse mapping*[4] (correlating target voxel to their respective source voxel before rotation) was applied. In the former case, it is possible for several source voxels to be assigned to the same voxel in the target volume. It is also possible that some voxels in the target volume might not be assigned at all, which leaves empty voxels. [25] The latter case, however, is not subjected to this inconvenience and produces a volume without empty voxels.

The script used for rotating a three-dimensional volume with a 3×3 rotation matrix $\boldsymbol{R}$ can be seen in Fig. 6. Instead of using for-loops this script works in a vectorized fashion.

A slightly faster alternative for rotating three-dimensional volumes is MATLAB's *imrotate3(...)* function. [24]. This function, however, could not be applied at the time when this work was drafted due to internal licensing issues.

2.3.4 Creation and export of cross sections

The result of the preceding code (Fig. 6) is a properly aligned volume, which enables the controlled extraction of radial cross sections along the component's longitudinal axis. Here, "properly" refers to the final state of the rotated volume. Ideally, one of the mask's eigenvectors is parallel to its rotational symmetry axis, and ends up congruently aligned with the reference eigenvectors. For the extraction, Bresenham's algorithm [26] was used, which is, in short, an efficient way of drawing lines between two points in a *discrete* raster[5]. The Bresenham algorithm for $\mathbb{R}^2$ can be described as follows: A line that is drawn between two points, which are on opposite sides on the circumference of a circle, can be approximated by a series of pixels that follow

$$\begin{aligned} &for |s| \leq 1, && for |s| > 1, \\ &x_i \in \mathbb{Z}, y \in \mathbb{R}, && y_i \in \mathbb{Z}, x \in \mathbb{R}, \end{aligned} \tag{14}$$

$$y(x_i) = s \cdot x_i + t, \qquad x(y_i) = s \cdot y_i + t, \tag{15}$$

$$y_i = \left[y(x_i)\right], \quad x_i = \left[x(y_i)\right], \tag{16}$$

$$\Delta_y = y_i - y(x_i), \quad \Delta_x = x_i - x(y_i), \tag{17}$$

with the square brackets [...] being the standard rounding functions to the closest integer, and

$$y_{i+1} = \begin{cases} y_i, & if (\Delta_y < 0 \wedge s > 0) \vee (\Delta_y > 0 \wedge s < 0), \\ [0.3cm] y_i + 1, & if (\Delta_y > 0 \wedge s > 0) \vee (\Delta_y = 0 \wedge s > 0), \\ [0.3cm] y_i - 1, & if (\Delta_y < 0 \wedge s < 0) \vee (\Delta_y = 0 \wedge s < 0), \end{cases} \tag{18}$$

with y_{i+1} being the next pixel added to the series for $|s| \leq 1$, and

$$x_{i+1} = \begin{cases} x_i, & if (\Delta_x < 0 \wedge s > 0) \vee (\Delta_x > 0 \wedge s < 0), \\ [0.3cm] x_i + 1, & if (\Delta_x > 0 \wedge s > 0) \vee (\Delta_x = 0 \wedge s > 0), \\ [0.3cm] x_i - 1, & if (\Delta_x < 0 \wedge s < 0) \vee (\Delta_x = 0 \wedge s < 0), \end{cases} \tag{19}$$

with x_{i+1} being the next pixel added to the series for $|s| > 1$. For lines with slope $|s| > 1$, x and y need to be interchanged in their functionality as argument and function value, respectively, as well as $s := s^{-1}$. Therefore, by inverting the coordinate system, it is guaranteed that for every new step along the argument's dimension the series of connected corresponding function values does not break.

In a final step, the resulting pixel series in $\mathbb{R}^2$ are expanded to the third dimension so that they form a vertical "wall" in a 3D (binary) volume $\mathbb{R}^3$ (Fig. 7) and are used as a reference mask to extract the radial cross sections along the component's longitudinal axis. Other line drawing algorithms were neither implemented nor performance-wise evaluated due to the fact that Bresenham's algorithm is comparatively quick and fulfills the desired functionality. However, it is worth mentioning that Bresenham's algorithm does not apply anti-aliasing, which, at low resolutions, gives the lines step-like appearance except when $|s|$ is zero or infinity. Wu's line algorithm [27] is an alternative to Bresenham that applies anti-aliasing. Since this involves additional arithmetical operations that would improve line appearance only at low resolutions but at the cost of a slower performance, we chose not to implement it.

3 Results

The outputs of the algorithm that we presented in the previous section are 2D cross-sectional images. The algorithm was tested on several volumes (see subsect. 2.1). The results from Vol1, Vol2 and Vol3 are almost identical to the respective cross sections from Vol5 (the synthetic volume), except for minimal deviations. Therefore, in order to avoid

[4] Inverse mappings are more efficient to implement than forward mappings and are used in numerous commercial implementations of spatial transformations [25].

[5] Simply using a conventional linear equation $y(x) = s \cdot x + t$ and rounding the resulting y only works in a satisfying manner when $|s| = 1$. If $|s| \neq 1$ and especially if $|s| \ll 1 \vee |s| \gg 1$ the results of y do not form a connected series of pixels, or voxels respectively.

duplicates, the radial cross sections from Vol1 to Vol3 are not shown here. Instead, only those from the brake disk volume Vol4 and the synthetic volume Vol5 are shown below. Each of the joining elements was sectioned 36 times at equally distanced angular steps. Each joining element is illustrated by one close-up cross section and 8, 10 or 18 additional cross sections, in order to save space. The results can be seen in Figs. 8, 9, 10, 11, 12, 13, 14, 15 and 16.

In addition to verifying the algorithm's automatic functionality, its performance was compared to two manual procedures, one virtual (in silico) and one physical. The former is non-destructive and was performed with the VolumeGraphics MAX 3.4 Software. The destructive manual procedure was not performed at this point due to a planned further use of the samples. Instead, skilled personnel were interviewed [28] and asked about how much time they spend on average on a single joining element. Destructively removing a single joining element from its structural surrounding (approx. 5 minutes per joint) to having it readily prepared for inspection under the microscope takes the average worker a total of around 25 minutes. The preparation involves carefully sawing the joint at its center, polishing it (approx. 5 minutes per joint) and finally treating it with 10-12 % caustic soda for 15 minutes [5] to remove residue. The chemical treatment can be applied to several joints at once, however, this process introduces a waiting period of at least 15 minutes, regardless of the number of joints. The results of this juxtaposition can be seen in Table 2. In order to evaluate the algorithm's performance under real conditions, the volumes were stored on a server, which is used by several people throughout the day. Therefore, in order to account for network traffic, the algorithm was executed ten times in a row for the automated approach.

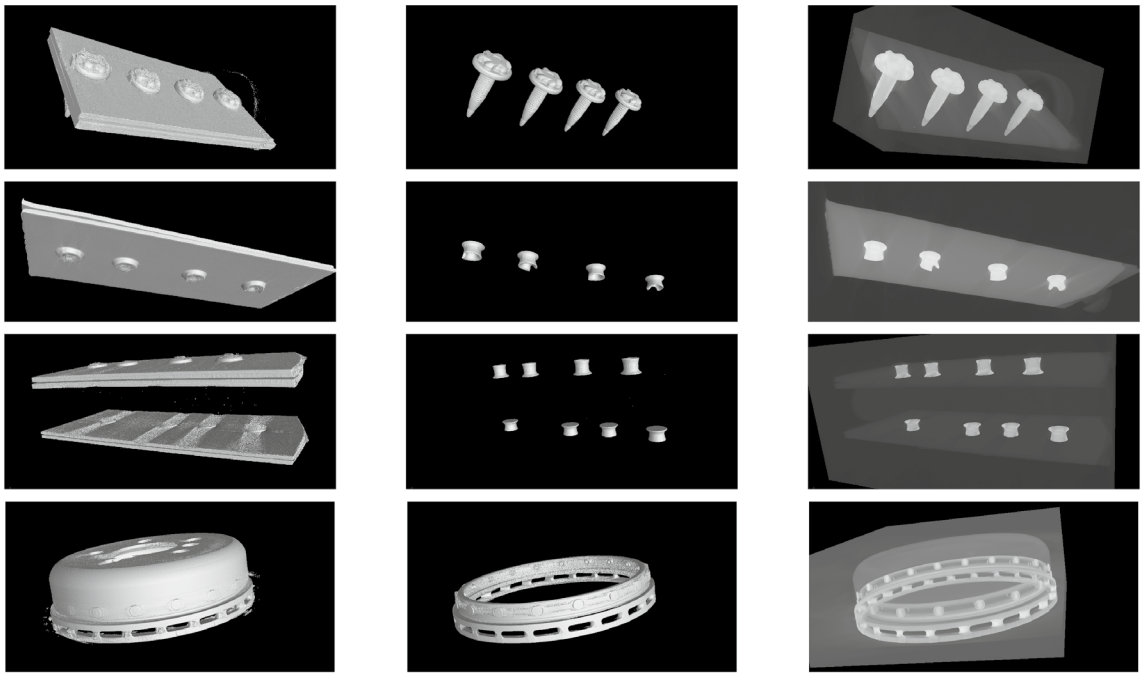
Both in-silico approaches used the same computer, which was equipped with 72 logic processors (Intel Xeon Gold 6154 CPU @ 3.00 GHz) and four AMD Fire Pro W8100 graphics processing units (GPUs). One as an internal GPU and three as external GPUs.

4 Discussion

This section follows the order of appearance of the results in the previous section and starts by discussing the mostly misaligned cross sections from the brake disk volume Vol4. A brief elaboration on potential improvement measures for such cases follows. Then, the results from the synthetic volume Vol5 are discussed. As previously stated in sect. 3, results from volume Vol1, Vol2 and Vol3 were not shown and therefore will not be discussed either. The results from Vol5 can instead be considered representative for them. In the end, the proposed automatic approach is compared with a manual approach with a commercially available software with respect to processing time.

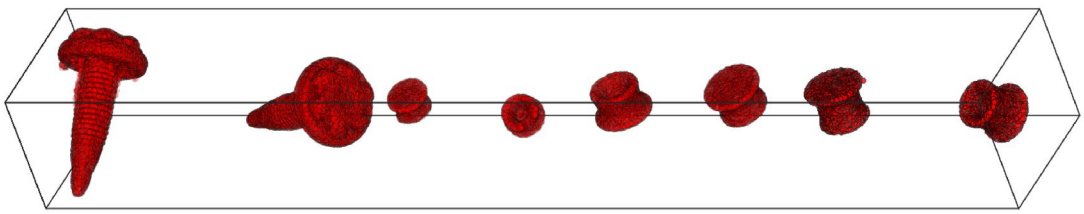
Cross sections from the brake disk volume Vol4. The algorithm has been applied to the scan of the modified brake disk (Vol4) in order to convey its limits. As stated in subsect. 2.1, additional components made of similar material to the target elements will negatively impact the algorithm's performance. This is simply due to the fact that the thresholding operation is not able to differentiate between relevant and irrelevant elements, but only between grayscale value peaks in a histogram. With that in mind, an erosion operation with a $(7 \times 7 \times 7)$ voxel structure element (SE) had to be applied to the binary volume after thresholding but before identifying connected components. This removed the cast iron ring in the scan, which is part of the remaining brake pot and connected to all eighteen rivets. It also removed voxels from the rivet. As can be seen in Fig. 8, after this user interaction the algorithm was able to isolate all eighteen rivets. However, due to artifacts and the erosion operation some rivets lost part of their ($\widehat{F}_{\ell}$) symmetrical shape around the longitudinal axis. Therefore, only some rivets were aligned properly, while all rivets appear to be smeared. This is a result of beam hardening, which is due to the polychromatic nature of the source spectrum. These quality degrading influences make a quantitative evaluation almost impossible. The seemingly arbitrary cut-out shape comes from the initial orientation and the final rotation, which is sometimes incorrect.

In order to improve the output quality of such samples, two things need to be realized. First, the data acquisition process or computed tomography, respectively, that was used needs to be optimized. The scan quality depends on various parameters, such as the voltage-current combination used, filter material and thickness, monochromaticity of the source, detector resolution, and source spot size, to name a few. If these parameters are not chosen carefully, the scan quality will be compromised. Another factor that influences the applicability of the proposed algorithm is the relative positioning of the sample. The closer the sample is to the source, and the further away the detector, the more details are visible due to a large geometric magnification on the detector. However, this comes at the cost of longer exposure times. Positioning the sample closer to the source limits the sample size and, therefore, also limits the number of joints per scanned volume. Together with the sample's geometry, the relative positioning has a significant impact on the discernible features in the resulting volume. Second, more advanced image processing operations are required in order to correctly discern the joints from unwanted structures, like the cast iron pot (Fig. 2, Figure 8), without compromising valuable voxels from joints themselves. A promising alternative to conventional image processing is machine learning algorithms.

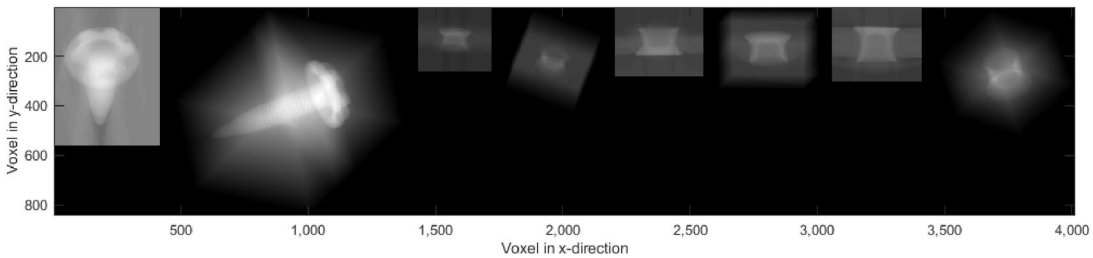


(a) Isosurface view (b) Steel elements (c) Maximum projections

Fig. 2 Physical samples that were used to test the algorithm. The left column shows a surface rendering of the complete sample, the middle column shows the steel and cast iron elements with aluminum virtually removed and the right column shows the samples with a certain degree of transparency. From top to bottom: Two aluminum plates joined by four flow-drilling screws (Vol1), two aluminum plates joined by four intentionally damaged self-piercing half-hollow rivets (SPRs) (Vol2), two sets of four SPRs joining two aluminum plates each (Vol3), brake disk with aluminum pot and cast iron disk (mostly removed) joined by eighteen flat head rivets (Vol4)



(a) The components visualized with MATLAB's 3D scatter plot function [14].



(b) Two-dimensional view of sample with projection summed up in the z-direction.

Fig. 3 Vol5, showing eight joining elements, two of each volume Vol1-Vol3. Every joint randomly rotated in order to generate this synthetic volume

Fig. 4 Excerpt from the script used for importing the scan data with MATLAB functions *fseek(...)* and *fread(...)*

```
% Input: Directory to the .vol and .pca file.
function varargout = main__import_data(varargin)
directory = varargin{1};
...
% For xSze, ySze, zSze.
pcrFileStruct = dir(strcat(directory, '*.', 'pcr'));
% For voxel data.
volFileStruct = dir(strcat(directory, '*.', 'vol'));
...
% Open vol-file
volFid = fopen(strcat(directory_vol, volFileStruct.name), 'r');
TotalBytes = volFileStruct.bytes;
BytesPerVoxel = TotalBytes/(xSze*ySze*zSze);
...
    for ii = 1:1:NrOfPackages
    ...
      % Use fseek to point to a certain byte from the file
      fseek(volFid,(ii-1)*BytesPerPackage,'bof');
      % Use fread for reading in a certain amount of bytes defined
      % by the 2nd input argument, starting from the byte pointed to
      % by fseek. Third input defines the byte depth. Therefore the
      % second argument must be divided by the bytes per volume.
      Content = fread(volFid, [1,BytesPerPackage/BytesPerVoxel], ...
        [dataType,'=>',dataType]);
      % Now properly reshape the read in package.
      ReshContent = reshape(Content, [xSze,ySze, SlicesPerPackage]);
      % Write to the final variable
      Imported(:,:,IndOld:IndOld+SlicesPerPackage-1) = ReshContent;
      fprintf('\n%d/%d Packages imported!\n', ii, NrOfPackages)
      IndOld = IndOld + SlicesPerPackage;
    end
    ...
% Output: Grayscale volume.
end % EOF
```

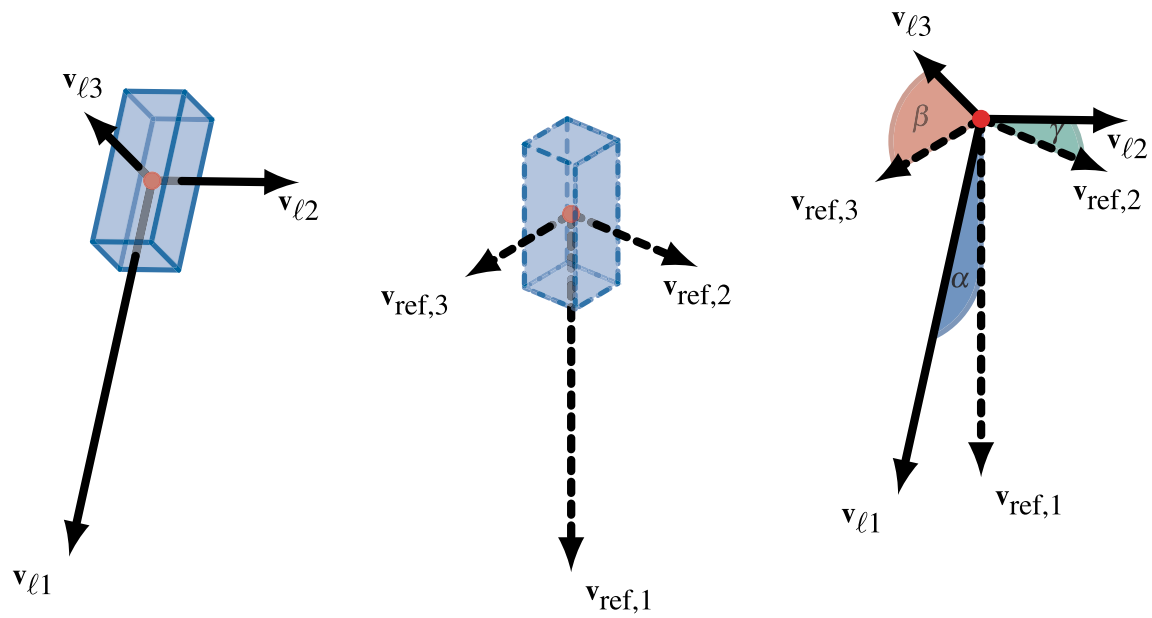


Fig. 5 Aligning eigenvectors with their respective reference vectors

Fig. 6 Vectorized script for rotations in three dimensions

```
function varargout = rotvol_vectorized(varargin)
% Input: 3D grayscale or binary volume, 3D rotation matrix.
SourceVolume = varargin{1}; % Grayscale or binary volume; to be
rotated.
R = varargin{2}; % 3x3 rotation matrix.
R = [[R, [0; 0; 0]];[0, 0, 0, 1]]; % Required expansion for
translation.
[x,y,z] = size(SourceVolume); % Dimensions of input volume.
% Matrix to translate center of rotation from origin to center of
source.
T = [1 0 0 (-x/2) ;...
     0 1 0 (-y/2) ;...
     0 0 1 (-z/2) ;...
     0 0 0    1   ];
ATMatrix = T\R*T; % Affine transformation matrix.
TargetVolume = zeros(x,y,z); % Preallocate rotated volume.
% Create subscript indices from linear indices.
[II, JJ, KK] = ind2sub([x,y,z], 1:x*y*z);
% Prepare subscript indices for matrix multiplication.
indAft = [II; JJ; KK; ones(1,x*y*z)];
% Rotate subscript indices from target to respective source indices.
indBef = round(ATMatrix\indAft); % Indices before rotation.
% Find rotated subscript indices (< 0) which do not exist in source.
tooSmall = any(indBef(1:3,:) <= 0, 1);
% Find rotated subscript indices (> x,y,z) which do not exist in
source.
x2Large = indBef(1,:) > x;
y2Large = indBef(2,:) > y;
z2Large = indBef(3,:) > z;
tooLarge = any([x2Large; y2Large; z2Large], 1);
OutOfBound = tooSmall \,|\, tooLarge; % Combine logical arrays.
% Delete all columns with indices outside of source boundary.
indBef(:,OutOfBound) = [];
indAft(:,OutOfBound) = [];
% Transform indivudual subscript indix rows to linear indices
Ind_lin_bef = sub2ind(size(Input), indBef(1,:), indBef(2,:), indBef
(3,:));
Ind_lin_aft = sub2ind(size(Input), indAft(1,:), indAft(2,:), indAft
(3,:));
% Use indices to transfere respective voxels from source to target.
TargetVolume(Ind_lin_aft) = SourceVolume(Ind_lin_bef);
TargetVolume = TargetVolume./max(TargetVolume(:)); % Normalize
volume.
varargout{1} = TargetVolume;
% Output: Rotated 3D grayscale or binary volume.
end % EOF
```

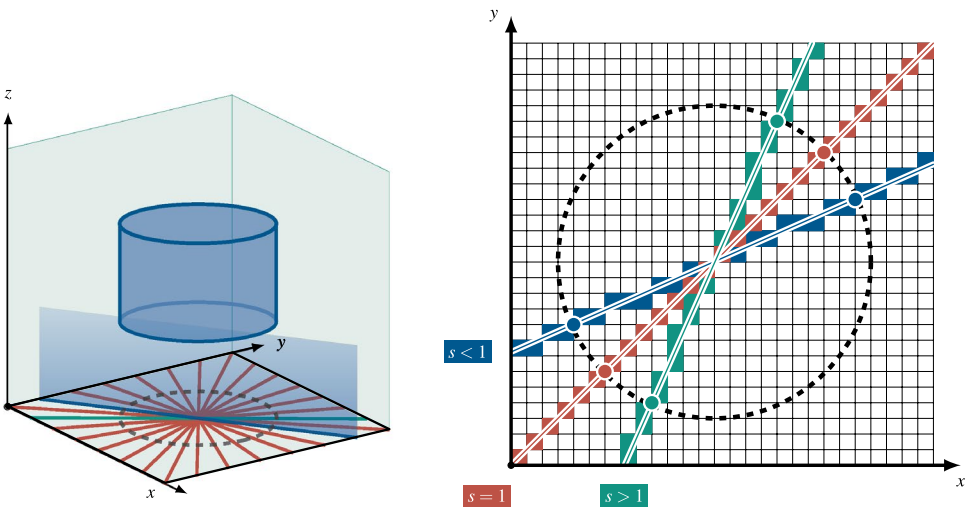


Fig. 7 The Bresenham algorithm. Left: Sectioning of the properly aligned elements. Right: Straight lines and the resulting pixel series to illustrate the cases shown in Eqs. 18 and 19

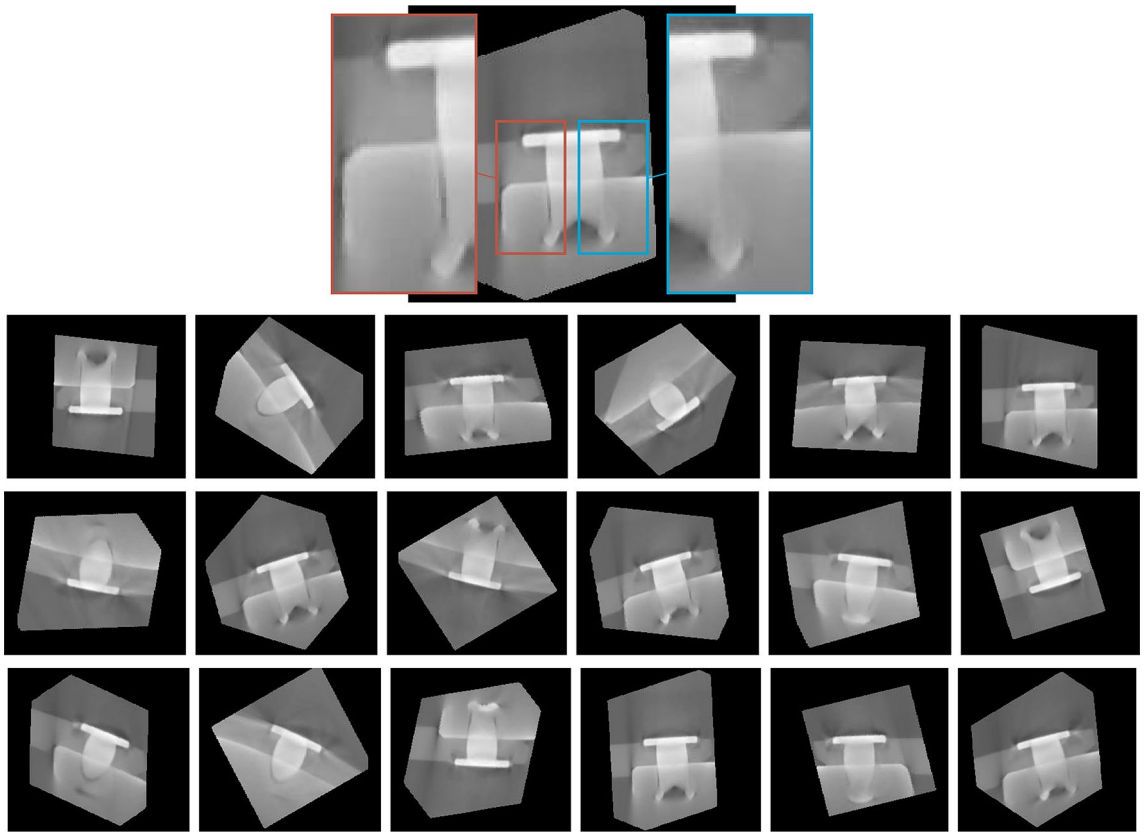


Fig. 8 Eighteen exemplary flat head rivets from the brake disk volume Vol4. Some were aligned correctly, others were not. Artifacts are present in the whole scan due to the cast iron pot and the steel rivets

Deep learning approaches, in particular, work very well for various images-related tasks (for example, see [29, 30]).

Cross sections from the synthetic volume Vol5. While some user interaction was required simply to isolate the flat head rivets and produce the rather unsatisfying cross sections in Fig. 8, no user interaction whatsoever was required to produce properly aligned and sharp cross sections in the case of flow-drilling screws, as can be seen in Figs. 9 and 10. On one hand, this is due to the advantageous geometry of the screws. In the context of principle component analysis (PCA), the elongated thread together with the conical screw tip are very discernible from the other two perpendicular components. On the other hand, the steel screw and aluminum plates provide much better contrast than is shown in Fig. 8. The above also applies to the flow-drilling screws of Vol1.

Probably the best illustration of the advantage CT has over a destructive 2D macro-section can be seen in Figs. 11 and 12. The defect[6] might not have been detected in a macro-section. Despite the negative impact on the symmetry of the rivet caused by the intentionally induced defect, the algorithm was still able to produce properly aligned and sharp cross sections with the defect visible. Since the joined structures are aluminum plates, the joints are easily detectable. The slight tilt is most probably the result of the PCA working on an asymmetrical shape. It is noticeable that the two rivets are mirrored horizontally. This is due to the sign ambiguity of the principle component analysis [31]. It arises from the fact that, in addition to the eigenvector $\boldsymbol{e}$ being a solution to $A\boldsymbol{e} = \lambda\boldsymbol{e}$, so is the eigenvector $-\boldsymbol{e}$. For this work and the further handling of the results, however, this mathematical inconvenience is not important. The above also applies to the rivets of Vol2.

The results in Figs. 13 and 14 show perfectly aligned cross sections. Here, again, the sign ambiguity flipped them upside down. It appears that the rivet's shaft almost punctures the bottom plate in every single one of the 22 images. If this is the case, it would be a failing criterion. However, this can only be assumed due to insufficient scan quality. Another failing criterion that might be assumed but not confirmed by visual inspection is the gap between rivet head and top plate, which can be seen in both magnified illustrations. Gaps like this make rivet joints

[6] Admittedly, the probability of this kind of defect actually existing is, in reality, very low. Nevertheless, it was quite easy to produce and to classify as a defect without having to measure its dimensions.

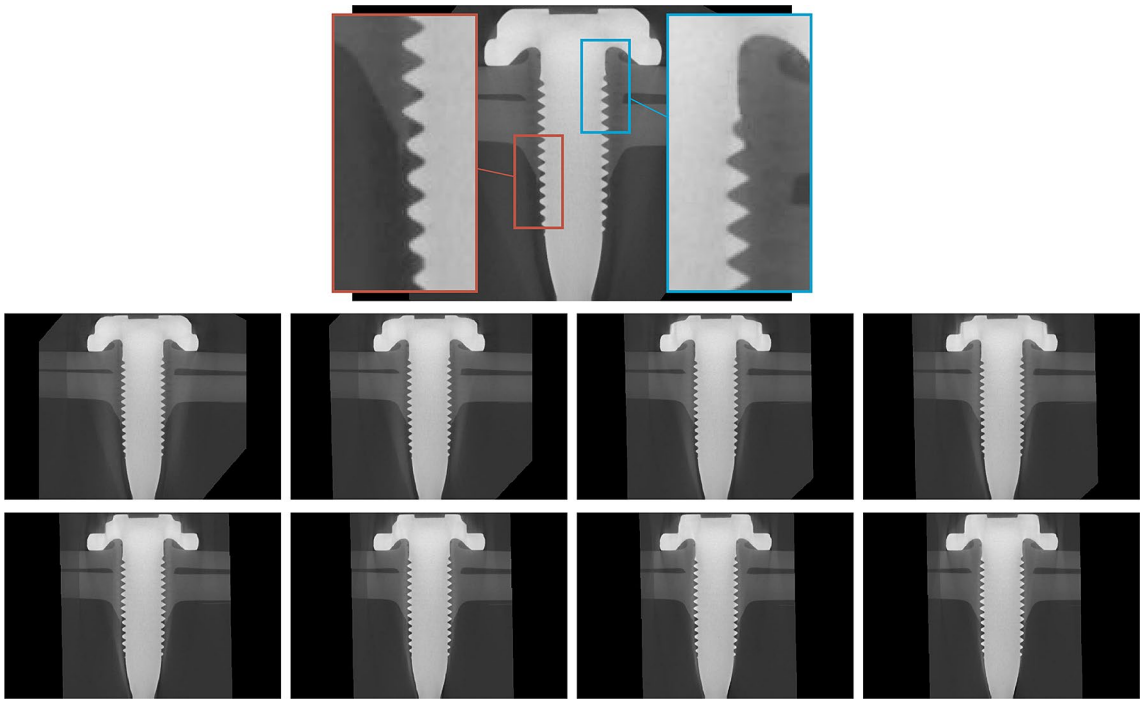


Fig. 9 First FDS from Vol5: It is clearly visible that a thread has been formed during the joining process (top). Eight different radial cross sections (bottom two rows)

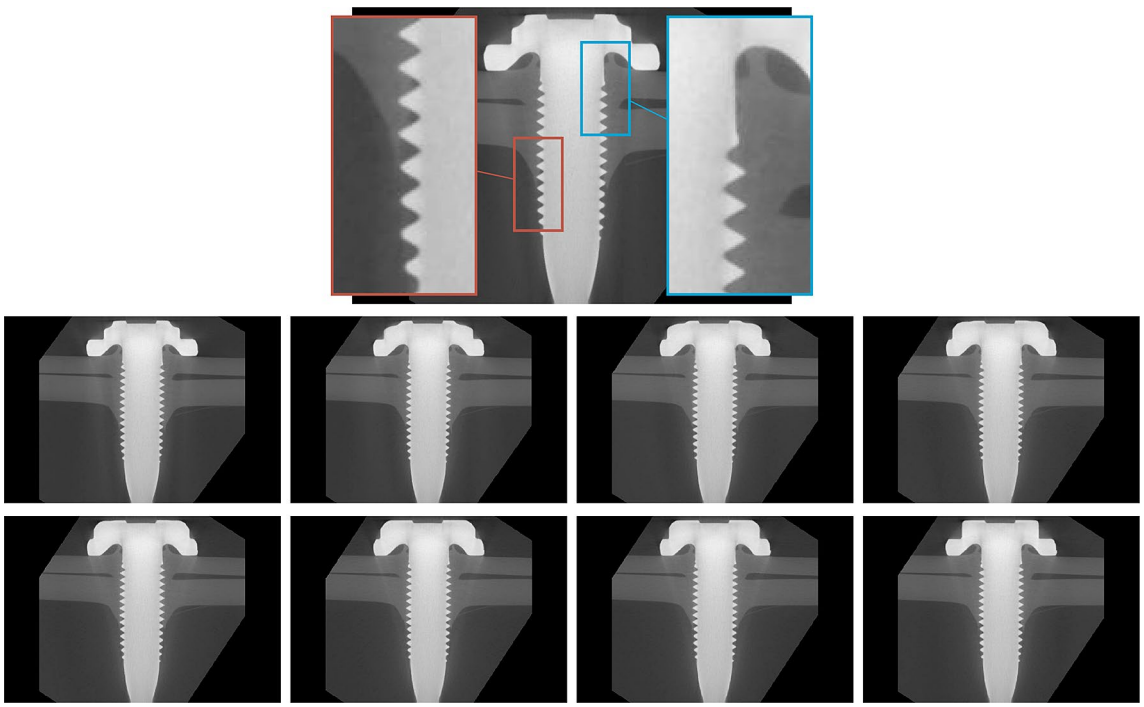


Fig. 10 Second FDS from Vol5: It is clearly visible that a thread has been formed during the joining process (top). Eight different radial cross sections (bottom two rows)

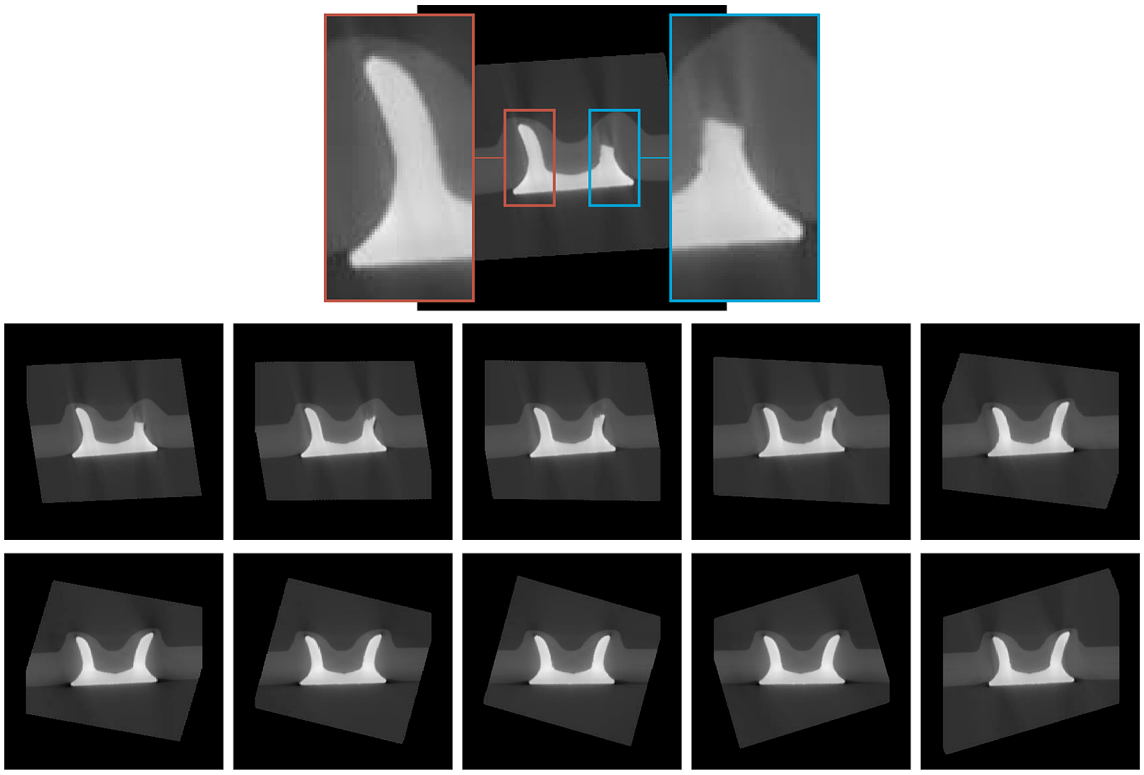


Fig. 11 First damaged SPR from Vol5: Rivet and joined plates are visible (top). Ten different radial cross sections with the defect being visible only in the first four (bottom two rows)

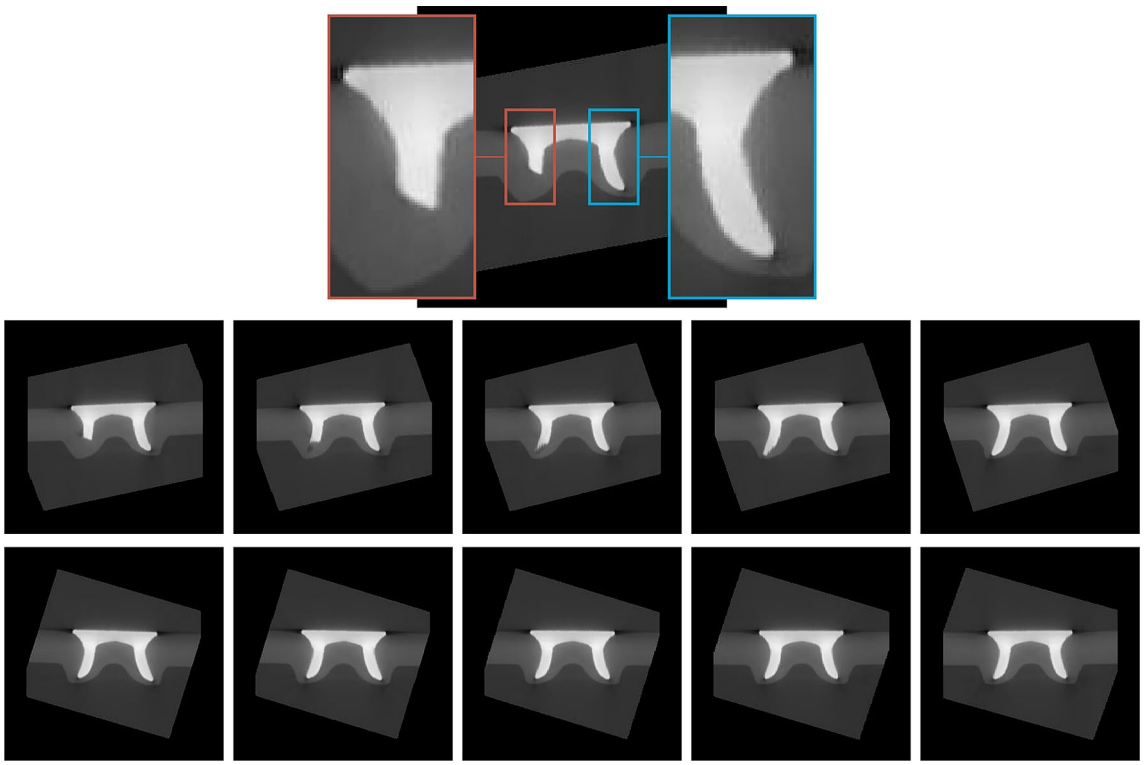


Fig. 12 Second damaged SPR from Vol5: Rivet and joined plates are visible (top). Ten different radial cross sections with the defect being visible only in the first three (bottom two rows)

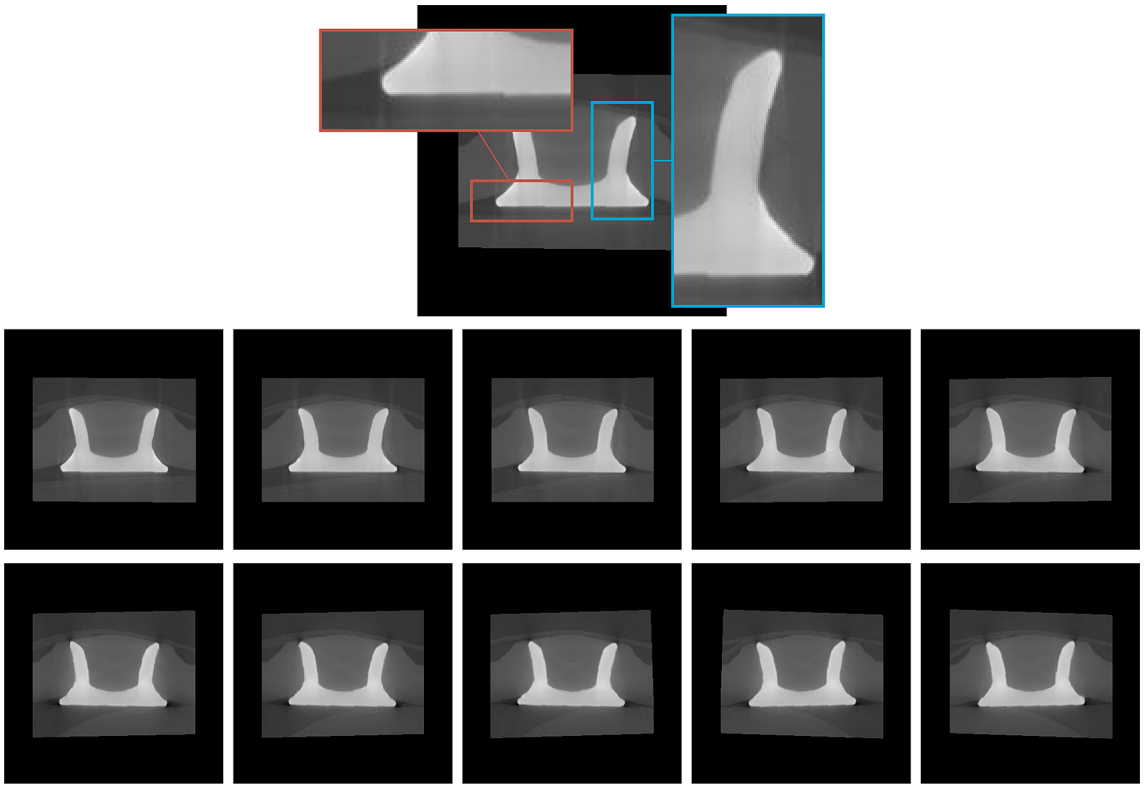


Fig. 13 First short SPR from Vol5: Rivet and joined plates are visible (top). Ten different radial cross sections (bottom two rows)

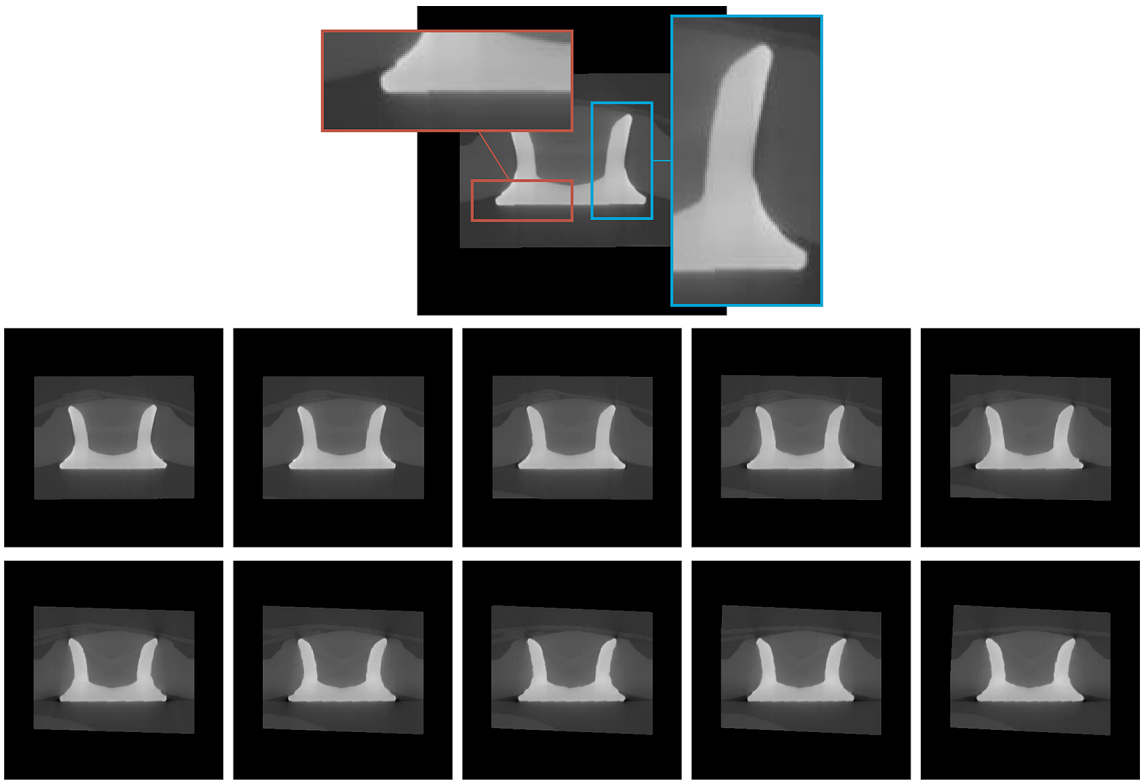


Fig. 14 Second short SPR from Vol5: Rivet and joined plates are visible (top). Ten different radial cross sections (bottom two rows)

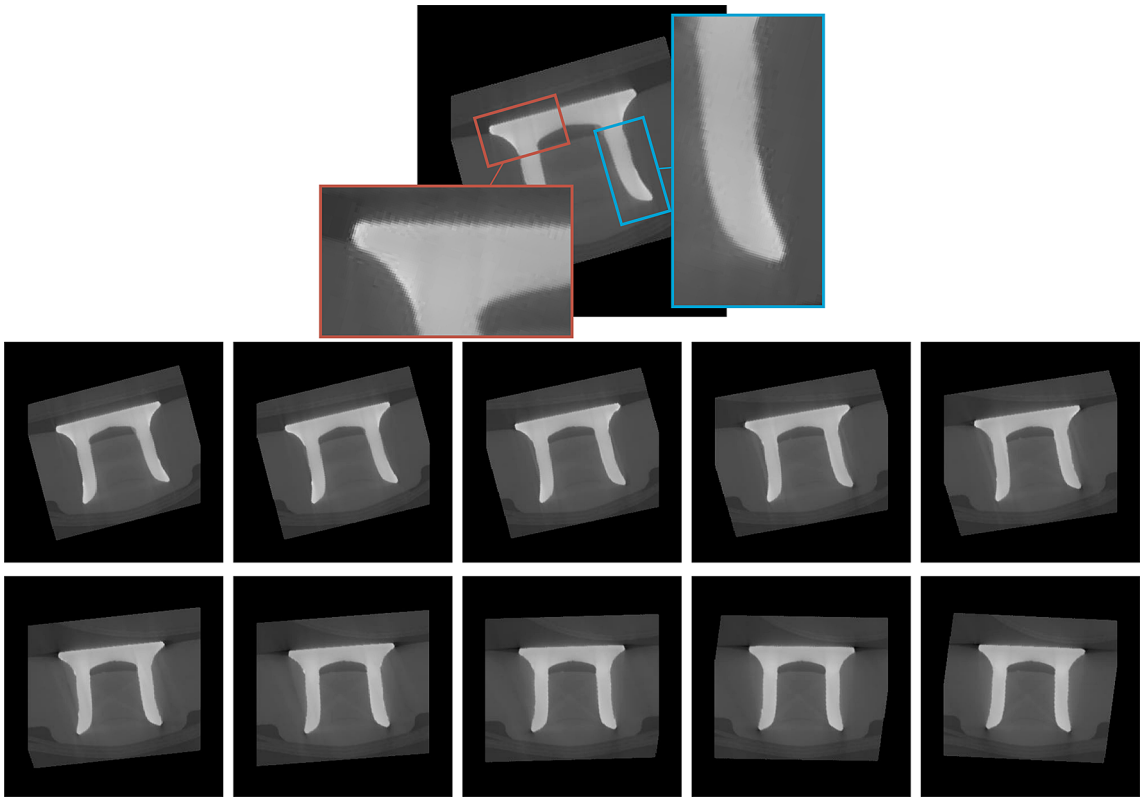


Fig. 15 First long SPR from Vol5: Rivet and joined plates are visible. The rivets exhibit subtle buckling at the shaft (top). Ten different radial cross sections (bottom two rows)

susceptible to corrosion and need to be prevented at all costs [2]. The above also applies to the short rivets of Vol3.

The rather pronounced tilt in Fig. 15 is immediately noticeable. This is caused by the longitudinal eigenvector not being aligned sufficiently with the rivet's longitudinal axis. Its geometric distribution of voxels was individually inspected in MATLAB. However, no obvious inhomogeneities were found. One long rivet from Vol3 was even aligned with a wrong axis. In Fig. 16, this tilt cannot be observed. One way to increase robustness for rivets with a round, flat head could be to incorporate the surface normal vector for the alignment operation. Another promising alternative is machine learning algorithms. Plans are in place to investigate the possibility of replacing the PCA-based rotation matrix with a rotation matrix that was determined by a neural network trained on covariance-rotation-matrix pairs. At this point, however, more investigation is needed in order to determine and eliminate the cause of this behavior.

While the cross sections produced in Fig. 16 can be used to evaluate the joint's quality, caution is advised when the same is done with the cross sections in Fig. 15. Since the alignment here is not performed properly, features that seem acceptable might actually not be acceptable at all. One example is the outward bending of the shaft. It seems to vary slightly along the images. Since it is largest in the top left image, only this one should be used for making a quantifiable analysis. The above also applies to the long rivets of Vol3.

Runtime analysis between the two non-destructive approaches. A comparison between the automatic approach and a virtual manual approach was also performed by one person. For this, the commercially available software VolumeGraphics MAX 3.4 was used. The evaluation process with this software included loading the volume, performing a surface determination, fitting a geometry element for properly aligning the sample and eventually exporting radial cross sections from every joint. The validity of this comparison is limited. This is due to the fact that the level of training with this software varies from person to person. Also, the used computing infrastructure allowed the commercial software to access a GPU for computationally intensive tasks. While there are hundreds of MATLAB functions that run automatically on GPUs [32], no attempt was made to quantify their individual impact on the algorithm's performance.

The manual approach with the commercial software has a very high probability of producing cross sections of high quality. This is due to the fact that conditions from

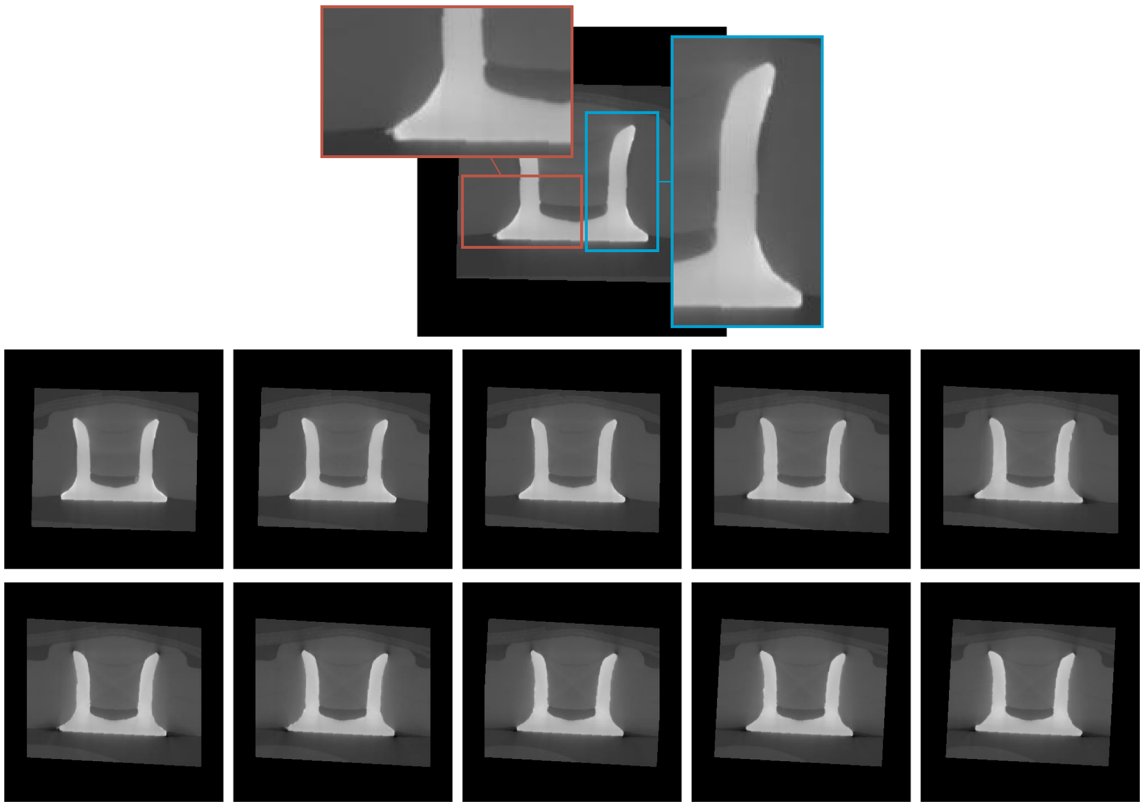


Fig. 16 Second long SPR from Vol5: Rivet and joined plates are visible. The rivets exhibit subtle buckling at the shaft (top). Ten different radial cross sections (bottom two rows)

Table 2 Results of runtime analysis of all volumes Vol1 to Vol5. Comparisons were drawn between an automated approach (in silico) and two manual approaches, one being non-destructive and the other being destructive

	Volume			Non-destructive		Destructive
	#Joints	Bytes	Scan time	In silico (autom.)	In silico (manually)	Physical (manually)
	[-]	[MB]	[min]	mean [min]±std [s]	[min]	[min]
Vol1	4×FDSs	4144	66	23.0 ± 64.6	≈ 9	100
Vol2	4×SPRs	9000	33	11.5 ± 2.6	≈ 9	100
Vol3	8×SPRs	46,995	133	$37.8 \pm 11.1^*$	≈ 33	200
Vol4	18×SPRs	6756	75	62.2 ± 7.3	≈ 35	450
Vol5	2×FDSs, 6×SPRs	1,925.7	-	25.8 ± 37.4	-	200

*The file was originally too large for the local memory, which is why the file had to be reduced by 80%. The time of size reduction is included

subsect. 2.2 do not apply to the commercial software because here, the user makes all the decisions and can select, segment or align features as he or she pleases. Here, for a skilled user the only limiting factor for the contextual task is the scan quality. Dissatisfying results like those in Fig. 15 were therefore easily avoided with the commercial software. The same argumentation holds for the results in Fig. 8. Here, too, properly aligned cross sections were produced with the commercial software. Nevertheless, the image quality remains a limitation.

If only the measured processing times for each volume are considered, the manual in-silico approach outperforms the proposed one on all volumes in terms of both speed and quality. Regarding quality, this statement is true since a skilled user can spend as much time as he or she wants on aligning the features and improving the image quality until an optimum is reached. The obvious advantage of the proposed algorithm over the commercial one is that it requires no user interaction beyond the user having to enter the file paths. It therefore can potentially be

applied and produce data around the clock, provided the aforementioned file paths are known before running the algorithm. This is especially desirable for working environments that facilitate inline inspection[7]. In addition, when the commercial software is used, some operations produce dead times, meaning the user has to wait for a varying amount of time before the next action can be performed. This creates a rather inefficient working environment, which is not the case with the functionality of the proposed algorithm.

The macro-functionality of VolumeGraphics is a promising alternative worth mentioning. However, this approach would require importing a previously generated and aligned reference file[8] of the specific processed joining element followed by a best-fit alignment operation between reference file and joining element. One drawback of this approach is a necessary user interaction: The user would have to select the macro with a suitable reference file before starting the automatic functionality. That means, only volumes with a constant composition of joining element could be automatically processed without further user interaction. Automatically evaluating volumes like Vol5, for example, would not be possible without creating a specifically tailored macro. Furthermore, there is an intrinsic weakness to using macros with reference files. It concerns the similarity between joining element and reference file: Due to differences in processing (e.g. varying sheet thickness and material but also setting force) the final joint might deviate from the shape of the reference file to a degree that jeopardizes their proper alignment.

Even though manually employing the commercial software's functionalities outperformed our automatic approach, it must be noted that manual work can only be performed at certain speeds and frequencies. The limits are usually set by work regulations or simply physiological factors. This is a limitation to which our approach does not fall victim.

The algorithm can reach its full potential especially in environments where the daily task is to analyze mechanical joining elements, like rivets or screws, both in large numbers and with a high throughput. Furthermore, one should refrain from cutting out and scanning single joining elements for reasons of workforce efficiency. Instead, a careful and systematic sample preparation should be considered. For example, one could try and mount several plates (like the ones in Vol1-3 as well as Vol5) on a vertical, tower-like sample holder and prepare a multi-scan CT project. Alternatively, an automated sample changer could be used.

Besides, we believe that a more powerful computational infrastructure as well as storing the required data locally can speed up the algorithm considerably. The latter is due to the elimination of network traffic impediment. However, no performance comparison between different hardware configurations was attempted.

However, it is important to keep in mind that the proposed algorithm's advantages over the manual approach can only be invoked in certain cases, e.g. Figs. 9, 10, 11, 12, 13 and 14. Based on this and the previous discussion, the added value of our work can be seen in contriving and writing a program that can replace manual, repetitive and destructive partial evaluations of mechanical joints, with an automated and non-destructive methodology. In addition, it was the concern of the authors to test the limits of classical image processing[9] as extensively as possible, in order to obtain a quantitative justification for a future machine learning approach. The authors are certainly aware that the methodology offers room for optimization. However, the added value generated by a coordinated workflow consisting of sample preparation, CT scan and our algorithm outweighs this in our opinion. Therefore, our algorithm should not be viewed as a general stand-alone remedy to manual evaluation tasks. Its methodology, and more precisely its inherent property of not needing individual reference files, should rather be implemented as a supplemental functionality to a more complete software package.

5 Conclusions

The proposed algorithm is intended to find and extract characteristic elements (joining elements) from a volumetric CT scan in a certain fashion. Once a potential element has been found, the algorithm extracts and rotates the element to an upright orientation and generates radial cross sections along the element's longitudinal axis. This functionality was demonstrated with different samples and the algorithm's performance was compared with two manual approaches, one being virtual and non-destructive, and the other being physical and destructive. In some cases, the proposed algorithm combined with computed tomography has been shown to provide a non-destructive, user-friendly and efficient alternative to destructive, manual approaches. The algorithm can

[7] The authors did not attempt a comparison with VGinLINE by Volume Graphics or similar inline software. Yet working environments with inline inspection are believed to benefit the most from algorithm functionalities like the ones we propose.

[8] For example an *.stl* file or a similar format that represents a surface mesh and can be imported.

[9] In the context of the previously discussed evaluation task (see sect. 1).

reach its full potential especially in environments where the daily task is to analyze mechanical joining elements like rivets or screws both in large numbers and with a high throughput. However, the proposed algorithm has limitations and should not be viewed as a general standalone remedy to manual evaluation tasks. Its methodology should rather be implemented as a supplemental functionality to a more complete software package.

The two major limiting factors for the algorithm are the samples themselves and the scan quality achieved. More advanced CT systems, as well as more advanced image processing algorithms, should be investigated in order to increase the algorithm's applicability and performance. Code optimizing strategies and memory usage were only partially dealt with in this work and should therefore also be investigated further. Together with the use of GPUs and/or more CPUs and with locally stored files, such strategies will most likely decrease the total run time and improve the quality of the resulting cross sections.

In a future work, this algorithm can be used to quickly produce unlabeled training data for various machine learning applications like pose estimation, joint dimensioning and semantic segmentation. The centered position of the elements in the cross sections produced by the algorithm makes the labeling process more comfortable, which is frequently the bottleneck in machine learning projects [33]. A semantic segmentation approach, for example, can learn to differentiate between joint and joined elements. Another useful application that would complete the workflow of the proposed algorithm would be a deep learning tool that is able to correctly measure the respective dimensions of a processed joint in order to establish whether or not it passes a quality check.

Acknowledgements We thank R. Gschneidinger from the BMW Plant in Dingolfing for providing the SPR and FDS samples, and U. Schulz from BMW Brake Disk Production in Berlin for providing the brake disk. We also thank M. Lieberwirth from the BMW workshop for preparing the brake disk for the CT, and D. Schuster from BMW Technology Material and Process Analysis for his support with preparing the CT scans.

Funding Open Access funding enabled and organized by Projekt DEAL. Necessary machinery, physical samples and software licences were provided by the Bayerische Motorenwerke AG, Germany.

Availability of data and material If required, all data including physical samples and CT scans can be provided. Please contact either author.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Code availability If required, all self-written scripts can be provided. Please contact either author.

References

1. Hahn O, Somasundaram S (2011) Handbuch Leichtbau: Methoden, Werkstoffe, Fertigung. Edited by Henning F and Moeller E. München: Carl Hanser Verlag
2. Li D, Chrysanthou A, Patel I, Williams G (2017) Self-piercing riveting-a review. The International Journal of Advanced Manufacturing Technology 92
3. ISO 13469:2014(en), Mechanical joining — Form-fit blind rivets and (lock) bolt joints — Specifications and qualification of testing procedures
4. ISO 12996:2013(en), Mechanical joining — Destructive testing of joints — Specimen dimensions and test procedure for tensile shear testing of single joints
5. Maier R, BMW Group Standard GS96001-2, 03/2010, BMW AG Normung: 80788 München
6. Carmignato S, Dewulf W, Leach R (2018) Industrial X-Ray computed tomography. Springer
7. Walz-Flannigan A, Brossoit K, Magnuson D, Schueler B (2018) Pictorial review of digital radiography artifacts. RadioGraphics 38:170038
8. Barrett J, Keat N (2004) Artifacts in CT: Recognition and avoidance. Radiographics 24:1679–1691
9. Lundström U (2014) Phase-contrast X-Ray carbon dioxide angiography. Doctoral Thesis. Royal Institute of Technology, Stockholm
10. https://www.ejot.de/Verbindungstechnik/Anwendungsbereiche/Automobilindustrie/Karosserie/FDS%C2%AE/p/VBT_FDS , company homepage, Accessed 11/22/2020
11. https://www.boellhoff.com/de-de/produkte-und-dienstleistungen/spezialverbindungselemente/stanzniete-rivset.php , company homepage, Accessed 11/22/2020
12. https://www.stanleyengineeredfastening.com/fasteners/self-piercing-rivets , company homepage, Accessed 11/22/2020
13. Henning F, Moeller E (eds) (2011) Handbuch Leichtbau: Methoden, Werkstoffe, Fertigung. Carl Hanser Verlag, München
14. https://de.mathworks.com/help/matlab/ref/scatter3.html , Mathworks.com Help Center, Accessed 12/08/2020
15. https://de.mathworks.com/help/matlab/ref/fseek.html , Mathworks.com Help Center, Accessed 12/08/2020
16. https://de.mathworks.com/help/matlab/ref/fread.html, Mathworks.com Help Center, accessed 12/08/2020
17. Kirz J, Jacobsen C (2009) The history and future of X-ray microscopy. J Phys Conf Ser 186:012001
18. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Sys Man Cyber 9(1):62–66

19. https://de.mathworks.com/help/images/ref/multithresh.html, Mathworks.com Help Center, accessed 12/08/2020
20. https://de.mathworks.com/help/images/ref/bwconncomp.html, Mathworks.com Help Center, accessed 12/08/2020
21. Rodrigues O (1840) Journal de Mathematiques, Vol 5, pp 380
22. https://de.mathworks.com/help/matlab/ref/cov.html, Mathworks.com Help Center, accessed 12/08/2020
23. https://de.mathworks.com/help/matlab/ref/eig.html, Mathworks.com Help Center, accessed 10/30/2020
24. https://de.mathworks.com/help/images/ref/imrotate3.html, Mathworks.com Help Center, accessed 10/30/2020
25. Gonzalez RC, Woods RE (2008) Digital image processing. Prentice Hall, Upper Saddle River, N.J
26. Bresenham JE (1965) Algorithm for computer control of a digital plotter. IBM Syst J 4(1):25–30
27. Wu X (1991) An efficient antialiasing technique. Comput Graph 25(4):143–152. https://doi.org/10.1145/127719.122734. ISBN 0-89791-436-8
28. Gschneidinger R (2019) Responsible for cold standard joining techniques at BMW Group, Dingolfing, 04/2019
29. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) : 770-778
30. Khan N (2020) A survey of the recent architectures of deep convolutional neural networks. Artif Intell Rev 53(8):5455–5516
31. Bro R, Acar E and Kolda TG (2008) Resolving the sign ambiguity in the singular value decomposition. J Chemometrics, 22:135-140. https://doi.org/10.1002/cem.1122https://prod-ng.sandia.gov/techlib-noauth/access-control.cgi/2007/076422.pdf, accessed 11/28/2020
32. https://de.mathworks.com/help/parallel-computing/run-matlab-functions-on-a-gpu.html, Mathworks.com Help Center, accessed 12/21/2020
33. Roh Y, Heon G, Wang SE (2019) A survey on data collection for machine learning. IEEE Transactions on Knowledge and Data Engineering 99:1-1