Visualization of deviations between different geometries using a multi-level voxel-based representation

Andreas Dietze
Fulda University of Applied Sciences
Leipziger Straße 123
36037 Fulda, Germany
andreas.dietze@cs.hs-fulda.de

Paul Grimm
Darmstadt University of Applied Sciences
Haardtring 100
64295 Darmstadt, Germany
paul.grimm@h-da.de

Yvonne Jung
Darmstadt University of Applied Sciences
Haardtring 100
64295 Darmstadt, Germany
yvonne.jung@h-da.de

ABSTRACT

We present an approach for visualizing deviations between a 3d printed object and its digital twin. The corresponding 3d visualization for instance allows to highlight particularly critical sections that indicate high deviations along with corresponding annotations. Therefore, the 3d printing thus needs to be reconstructed in 3d, again. However, since the original 3d model that served as blueprint for the 3d printer typically differs topology-wise from the 3d reconstructed model, the corresponding geometries cannot simply be compared on a per-vertex basis. Thus, to be able to easily compare two topologically different geometries, we use a multi-level voxel-based representation for both data sets. Besides using different appearance properties to show deviations, a quantitative comparison of the voxel-sets based on statistical methods is added as input for the visualization. These methods are also compared to determine the best solution in terms of the shape differences and how the results differ, when comparing either voxelized volumes or hulls. The application VoxMesh integrates these concepts into an application and provides the possibility to save the results in form of voxel-sets, meshes and point clouds persistently, that can either be used by third party software or VoxMesh to efficiently reproduce and visualize the results of the shape analysis.

Keywords

3D Object Comparison, Difference Visualization, Shape Similarity, Digital Twin, Voxel-based Modeling

1 INTRODUCTION

The acquisition, analysis and processing of 3d data based on depth data is still a current research area, as examples in the fields of visual computing like digital construction monitoring [DGJ20] show. This correlates with the fact that due to advances in augmented reality (AR), 3d sensing, and 3d scanning, many new mobile devices, such as the Samsung S21 Ultra or Apple's iPad, have advanced technologies for acquiring 3d data integrated into their product lineup. In terms of mobile devices, the technologies used (e.g., SfM, ToF, LiDAR) primarily serve to enrich a real scene with digital content, but are also used in the field of 3d reconstruction, as can be seen in the example of Microsoft's HoloLens mixed reality headset or Apple's LiDAR sensor, which are used, for example, in the context of digital construction monitoring and surveying [WWWH21, DGJ21].

In addition, more and more affordable and professional devices or systems in the field of 3d reconstruction are appearing on the market (e.g., from Shining 3D), which are suitable for high-resolution and detailed 3d reconstruction of smaller parts (e.g., gear wheel, EinScan - SP1[1]) to medium sized objects (e.g., car door, EinScan HX2[2]). The resulting 3d data by such scanners, for example, can be used for a comparison between the 3d geometry of the planning data and a 3d reconstruction of a printed object (digital twin) in terms of their shape similarity. This allows the localization and measurement of deviations between these two geometries and could be used as a non-destructive testing method [WZL$^+$20] in the field of additive manufacturing processes like 3d printing.

One of the problems that impacts the overall quality or functionality of the printed object is the 3d print warping of individual areas. These deformations are caused by the material shrinkage due to heating (expansion) and cooling (contraction) of the material. Another source of deformations is the use of support mate-

[1] Shining 3D EinScan-SE: https://www.einscan.com/desktop-3d-scanners-de/einscan-sp/

[2] Shining 3D EinScan HX: https://www.einscan.com/multifunktionaler-3d-scanner/einscan-hx/

https://www.doi.org/10.24132/CSRN.3301.18

rial that can be wrongly placed or even missing, which obviously negatively influences the final shape of the printed object.

The main contribution of this paper is a novel method for visualizing and highlighting deviations between 3d printed products and their 3d reconstruction based on a multi-level voxelization (MLV) of topologically different geometries. This approach utilizes volumetric mesh voxelizations (including interior voxels) as well as voxelized mesh hulls (lacking interior voxels), where the latter allows higher grid resolutions. The MLV also allows determining a quality measure with statistical methods. The measurement of found deviations allows to classify if deviations are within a given tolerance, while the determined value of the shape similarity reflects an overall status of the deviations and could be used as a threshold value.

The benefit is to visualize differences in an early stage between the 3d planning data and its 3d reconstruction of the printed object to avoid follow-up costs in mass production or special spare parts. Visualizing deviations and critical sections like missing parts or strong deformations can help in terms of quality assurance as a non-destructive testing method, but also can help to adjust the printing settings (e.g., missing support material or layer height) and to ensure the correctness of printed spare parts.

2 RELATED WORK

For visualizing deviations between topologically different geometries via voxel-based representations, related work in the fields of object registration, voxelization, shape analysis and geometric similarity have to be considered. Efficiently highlighting the results of shape analysis is a broad field by itself and several approaches and tools already have been developed (cp. e.g. [OGBS06]).

In Novotni et al. [MR01] a method is described, where objects are first superimposed and aligned based on their center of mass and the eigenvectors before a similarity of the 3d objects is determined on the basis of their geometric properties (geometric similarity). The determination of geometric similarity was realized via distance fields to calculate offset-hulls, which provide information about overlapping areas of two volumes and are illustrated in the form of distance histograms.

Another method for measuring the similarity of 3d models is described in Chen et al. [CHL$^+$17], in which a similarity measurement is performed on the basis of skeleton trees. For this purpose, the skeleton trees are created based on the topology of the 3d model's skeleton, so that the topological and geometric properties of the 3d models are represented and compared using the tree structure.

Furthermore, in Doboš et al. [DFFW18] a method is presented that detects differences between 3d models in the screen space and visualizes them for the user. To detect differences, various data such as color, depth, normals and texture coordinates are compared within screen space.

A method with which 3d planning data for an additive manufacturing process is analyzed for its geometric and mechanical properties prior to the manufacturing process is presented in Rupal et al. [RMWQ19]. This is achieved by converting the sliced print data of the 3d model back into a CAD model in a reverse process to enable optimization of the print settings.

Furthermore, in the past years neural networks are increasingly used for a geometric comparison of 3d models, even if they are mostly used for object recognition and classification in the sense of object similarity [KWL20, NZL$^+$20, LEAM$^+$19]. A comparison of CAD planning data and its components using machine learning is described in Bickel et al. [BSSW21]. Here, the 3d planning data for new components are compared with the 3d planning data for existing components.

3 CONCEPT

Our proposed concept for visualizing deviations based on shape analysis and the comparison between the original 3d planning data and the 3d reconstruction of the 3d printed output is based on the following steps:

1. Data acquisition: use 3d planning data to print objects and scan the printed output to provide a mesh-based 3d reconstruction
2. Shape analysis: provide similarity indices and voxel sets for visualization
   (a) Pre-processing: positioning (superimposition) and alignment of the acquired data sets
   (b) Multi-level voxelization (MLV): both data sets are voxelized separately and written to a file
   (c) Shape similarity: compute similarity based on common statistical methods of the resulting voxel sets
3. Visualization: found deviations are visualized using a voxel-based or mesh-based representation

Figure 1 contains the pipeline of the whole process to summarize the complete procedure described in the next chapters.

4 DATA ACQUISITION

To provide data sets used as input for our prototypical implementation of the concept, we printed and reconstructed four objects (teapot, cat, frog, gearwheel). In

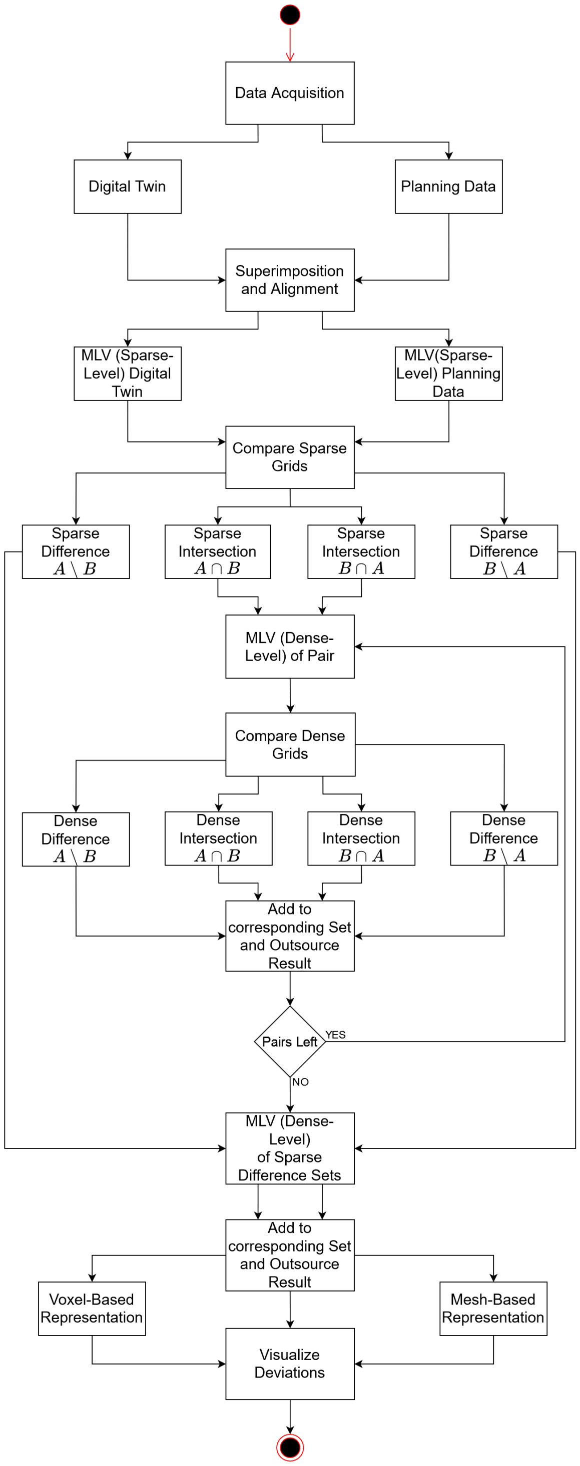

Data Acquisition
Digital Twin
Planning Data
Superimposition and Alignment
MLV (Sparse-Level) Digital Twin
MLV(Sparse-Level) Planning Data
Compare Sparse Grids
Sparse Difference $A \setminus B$
Sparse Intersection $A \cap B$
Sparse Intersection $B \cap A$
Sparse Difference $B \setminus A$
MLV (Dense-Level) of Pair
Compare Dense Grids
Dense Difference $A \setminus B$
Dense Intersection $A \cap B$
Dense Intersection $B \cap A$
Dense Difference $B \setminus A$
Add to corresponding Set and Outsource Result
Pairs Left
YES
NO
MLV (Dense-Level) of Sparse Difference Sets
Add to corresponding Set and Outsource Result
Voxel-Based Representation
Mesh-Based Representation
Visualize Deviations

Figure 1: Illustration of the system pipeline.

addition, for one object (teapot) also manipulated planning data was used as reconstructed data, which made it easier to intentionally insert errors for testing.

As 3d printer, a Creality CR-10 V3 was used. The printer filament consisted of (green) PLA (Polylactic Acid), which is a common and widespread material used for 3d printing. The objects were printed with a layer height of 0.28mm at an extruder temperature of 210° and a print bed temperature of 50°.

A model-based 3d reconstruction of the printed objects was carried out by a Shining 3D EinScan SE scanner and the included Software EXScan[3] V.3.0.0 without any further mesh optimizations. This reconstruction represents the digital twin of the constructed planning data that was used for the comparison with the 3d planning data. Figure 2 contains the 3d planning data mesh as well as the corresponding 3d reconstruction from the printed object. Both, the planning data and the reconstruction differ in number of geometric primitives and their topology. The 3d reconstruction shown on the left consists of 93,627 vertices and 97,824 triangles. The planning data on the right consists of 10,206 vertices and 6,320 triangles.

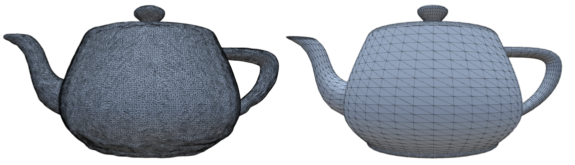


Figure 2: Topologically different data sets. Left: 3d reconstruction (93,627 vertices, 97,824 triangles). Right: original 3d planning data (10,206 vertices, 6,320 triangles).

5 SHAPE ANALYSIS

Pre-processing

For our proposed multi-level voxelization and shape analysis, an initial overlay regarding the position and orientation of the meshes to be compared is necessary, since the shape similarity is based on a statistical comparison of the different voxel sets resulting from the voxelization process (superimpositions and deviations, or intersection and difference sets). If the two objects are not initially superimposed and aligned, the objects are geometrically registered with the help of a semiautomatic approach using a principal component analysis (PCA) based on their center of mass and aligned using their eigenvectors. In the majority of our test cases, manual adjustment of the overlay and alignment (see Figure 3) was required following the application of the PCA due to differences in topology. As the focus of this paper is on comparison rather than alignment, this is sufficient but could be automated in the future.

Multi-Level Voxelization

The voxelization of the 3d planning data and the 3d reconstruction is performed in multiple voxelization steps

[3] EXScan: `https://www.einscan.com/einscan-software/exscan-pro-software-download/`

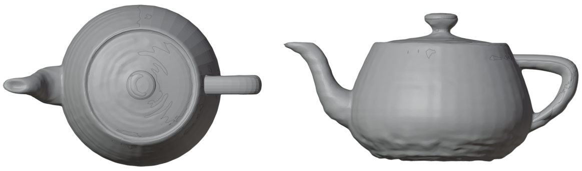


Figure 3: Superimposition and alignment of the planning data and 3d reconstruction of the printed object.

that differ in their implementation concerning the intersection of the voxel with the meshes. This is in relation with the voxelized results, that either can consist of voxelized hulls or volumes of the geometry. Regardless of the chosen voxelization result, the same input data provided by the Shining 3D EinScan SE is used.

In both cases, the first level consists of a sparse voxelization of the mesh hulls. This, on the one hand, is used as a space partitioning method to divide a predefined local space (e.g., 10cm x 5cm x 10cm) for each of the two meshes to be compared, resulting in a sparse 3d voxel grid for each mesh, in which a voxel either overlaps with the mesh or not. On the other hand, a comparison between the resulting voxel sets provides first information about critical sections, since there are deviations detected even at a sparse voxel resolution in the first level voxelization.

To determine the voxelized hull by an intersection of a voxel and the mesh, Unity's physics engine[4] was used. In order to provide the voxelized volume, the intersection logic is performed via raycasting, where six rays starting from the center of a voxel check whether they collide with a surface of the mesh or not. The maximum number of voxels $v \in \mathbb{N}$ results from the length a_1, height a_2, and width a_3 of the local space to be measured and the parameterized voxel size $s \in \mathbb{R}^+$, which divides $a_1 \cdot a_2 \cdot a_3$. In Figure 4 a first level voxelization of the 3d reconstruction is shown. On the left, the voxelization consists only of the object's hull. On the right, the result consists of the voxelized hull (gray) and the volume (blue) of the mesh.

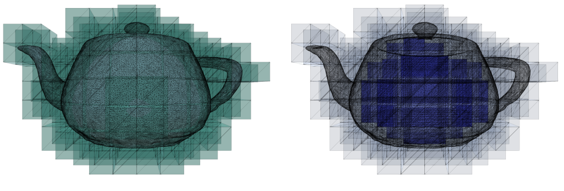


Figure 4: First level voxelization with a sparse grid resolution of 0.5cm of the 3d reconstruction. Left, voxelization of the mesh hull. Right, voxelization of hull and interior of mesh.

At the second level voxelization, the procedure is repeated for each sparse voxel cell. In case of the hull voxelization, every voxel is divided into a dense 3d voxel grid and again, an intersection with the mesh and the voxels is performed with Unity's integrated physics engine that provides a high resolution voxelization, depending on the used dense voxel size. For the voxelized volumes, the intersection is performed using raycasting. Due to the cubic complexity of the voxelization process, the number of voxels, the runtime of voxelization and comparison, the memory consumption and a subsequent real-time visualization depend on the voxel sizes used for both (sparse and dense) 3d voxel grids and increase with higher accuracy (higher resolution due to smaller voxel size).

In the worst case, the number of voxels to be processed is larger than the available main memory. In terms of memory and runtime, the second level voxelization benefits from the first level voxelization, since here only the voxels resulting from the first level are voxelized in the dense resolution. In addition, each sparse voxel of the planning data result is compared one by one with the corresponding voxel of the 3d reconstruction's voxelization result to limit the memory consumption to those two cells instead of voxelizing all sparse cells at once.

In detail, the corresponding sparse cells are first voxelized using the dense 3d grid and either the physics or raycast procedure for the voxel-mesh intersection, which either results in a subset of the hull or volume of the meshpart enclosed by the sparse voxel. In the next step, both sparse voxels containing the second level voxelization result of those two cells are centered and each dense voxels are compared using the L2-norm detecting their euclidian distances. This determines which dense voxels intersect with both the planning data mesh and the reconstructed mesh (intersection set) and those that do not (difference sets).

Following this, the result in form of the determined voxel sets (intersection set and or difference sets) are written into a file and the used memory will be freed for the next sparse voxel pair. Sparse voxel cells that are either not overlapping with the planning data or 3d reconstruction are voxelized in the final step for each sparse voxel set individually and the second level voxelization result is added to the corresponding dense set (see Figure 5) and also written to the file, which, in addition, provides a persistent result of the whole process.

Shape Similarity

The determination of the shape similarity is based on the voxel sets resulting from the MLV. For the 3d planning data they consist of two voxel sets, one resulting from the sparse first level voxelization A_s, and the other dense voxel set A_d, resulting from the second level voxelization. Equivalent to this, the voxelization of the reconstructed results in B_s for the sparse set and B_d for the dense set.

[4] `https://docs.unity.com`

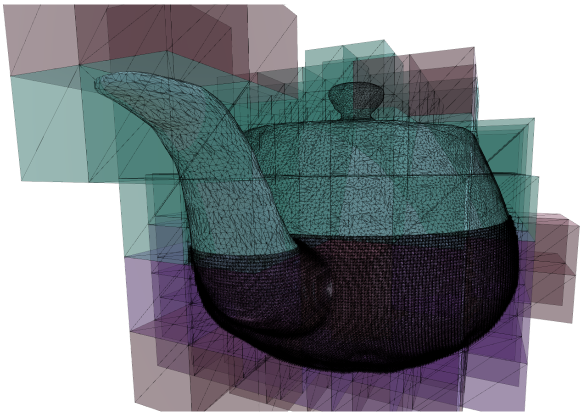


Figure 5: Visualization of the MLV process for the teapot's digital twin using a sparse resolution of 0.5cm and a dense resolution of 0.25mm. Green voxels are superimposed both with the meshes of the 3d reconstruction and the planning data respectively (intersection); red voxels do not overlap (difference set) and violet voxels have already been transferred to the dense voxel grid and compared in pairs. In a final step, the difference sets are voxelized.

A comparison between the two sparse voxel sets A_s and B_s is based on an overlay resulting in an intersection set $A_s \cap B_s$ that ideally corresponds to the union $A_s \cup B_s$ with a maximum degree of shape similarity, which also implies that the number of voxels in A_s equals the number of voxels in B_s. If the sets A_s and B_s have a different number of voxels, this results in the two difference sets $A_s \setminus B_s$ and $B_s \setminus A_s$. Moreover, this already provides information about critical sections since there are deviations detected between both meshes at the first level voxelization even without further approximation.

During the analysis of an overlapping voxel pair of $A_s \cap B_s$, all containing voxels of the two voxel sets A_d and B_d are checked for overlapping, resulting in the intersection $A_d \cap B_d$ and the two difference sets $A_d \setminus B_d$ and $B_d \setminus A_d$. Based on the quantity of those dense voxel sets, a shape similarity is determined by the use of established statistical methods providing a similarity coefficient using the Dice Index (DI) [ZWB+04], the Jaccard Index (JI) [FI18], and the Kulczynski Index (KI) [ZAB+16], where every similarity coefficient is in the interval $[0,1]$, while a higher value expresses a higher similarity.

The DI results of twice the quantity of the intersection set $|A_d \cap B_d|$ and the two difference sets $|A_d \setminus B_d|$ and $|B_d \setminus A_d|$.

$$DI = \frac{2|A_d \cap B_d|}{2|A_d \cap B_d| + |A_d \setminus B_d| + |B_d \setminus A_d|} = \frac{2|A_d \cap B_d|}{|A_d| + |B_d|} \tag{1}$$

The JI describes the cardinality of the intersection set $|A_d \cap B_d|$ and the union set $|A_d \cup B_d|$.

$$JI = \frac{|A_d \cap B_d|}{|A_d| + |B_d| - |A_d \cap B_d|} = \frac{|A_d \cap B_d|}{|A_d \cup B_d|} \tag{2}$$

The KI is defined by the intersection set $|A_d \cap B_d|$ and the two difference sets $|A_d \setminus B_d|$ and $|B_d \setminus A_d|$.

$$KI = \frac{1}{2}\left(\frac{|A_d \cap B_d|}{|A_d \cap B_d| + |A_d \setminus B_d|} + \frac{|A_d \cap B_d|}{|A_d \cap B_d| + |B_d \setminus A_d|}\right) \tag{3}$$

Results and Evaluation

All object pairs consisting of the planning data and the reconstructed meshes to be compared are located in two predefined local spaces (e.g., 10cm x 5cm x 10cm), which both are divided into a sparse and dense voxel grid by the MLV using the initially defined sparse and dense voxel size. Table 1 contains an overview of the voxel sizes used and the resulting resolution of the sparse and dense grid. Especially the size of a dense voxel is relevant here, since it corresponds to the unit size of a voxel in the voxelized meshes.

Table 2 contains the result of the comparison based on the voxel sets (hulls) of the 3d reconstruction (A) and 3d planning data (B). This consists of the determined number of voxels per mesh, the determined intersection, differences and union sets as well as the found shape similarity, based on the aforementioned Equations 1 (DI), 2 (JI), and 3 (KI).

It is also worth mentioning, that the first teapot in the table represents the shape analysis between the manually deformed digital twin (teapot without spout and handle, see Figure 7) and its planning data, whereas the second represents the 3d reconstruction of the printed teapot. The other three objects (cat, frog and gearwheel) are representing additional results from the shape analysis based on the MLV presented in Chapter 5.

Table 3 contains all objects using a volumetric MLV compared to the objects in Table 2, in which the voxelization results consists of the voxelized hulls. Noticeable is the similarity of the results between the DI and KI. Compared to the DI and KI, the JI results in a slightly smaller similarity coefficient. The voxelization of the planning data in the respective resolution is regarded as ground truth. A comparison of this data set with itself results in a shape similarity index of 1.0 concerning all statistical methods used, regardless of the chosen grid settings. Consequently, no deviations were identified either.

Comparing the similarity coefficients of Table 2 (voxelized hulls) and Table 3 (volumetric voxelization), it is noticeable that the results based on the hull generally give a lower value compared to the results based on the volumes. In addition, the results are lower applying a higher resolution using the hulls compared to the volumes, where the results only slightly differ using different resolutions.

As test system, a Ryzen 9 3900X CPU with 32 GB DDR-IV, an NVidia RTX 3080 GPU on a 970 Evo Plus M.2 SSD, was used. Table 4 shows the required runtimes for the voxelization process including the determination of the shape similarity for the hulls and volumes of the teapot (second object in Table 2 and first object in Table 3).

6 DEVIATION VISUALIZATION

The visualization of the detected deviations is based on the different voxel sets resulting from the MLV. These are either visualized directly (voxel-based representation) or the localized deviations are transferred to the input meshes (mesh-based representation). For both representations the following simple color scheme exemplarily is used:

BLUE: intersection $A_d \cap B_d$
RED: difference $A_d \setminus B_d$
YELLOW: difference $B_d \setminus A_d$

Voxel-based Representation

The voxel-based representation of all computed deviations is based on the intersection set and both disjoint difference sets between the voxel sets A_d and B_d, which can be individually visualized. In Figure 6, the incorrect teapot on the left is missing its spout and handle but also has a larger knob on the cover. It is representing the 3d reconstruction data that has been manually modified to provide some obvious deviations. The teapot on the right represents the 3d planning data. As can be seen, too, both meshes differ in their topology to simulate a mesh-based 3d reconstruction.

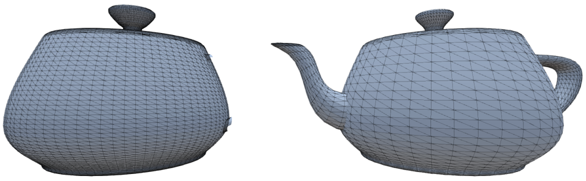


Figure 6: Left: 3d reconstruction with obvious deviations. Right: 3d planning data.

The found deviations are visualized individually on top of their corresponding mesh in Figure 7. On the left, deviations concerning the larger knob of the 3d reconstruction are shown, which represent an error that is located outside of the 3d planning data. In terms of the planning data, errors are located at the spout and handle but also concerning the smaller knob, as can be seen on the right. In the middle, the intersection set is shown.

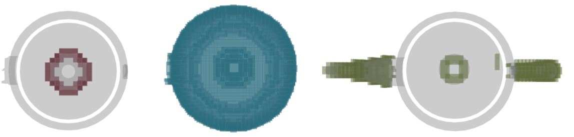


Figure 7: Left: difference set $A_d \setminus B_d$ (red). Center: intersection set $A_d \cap B_d$ (blue). Right: difference set $B_d \setminus A_d$ (yellow). The difference sets $A_d \setminus B_d$ and $B_d \setminus A_d$ are superimposed with their corresponding mesh.

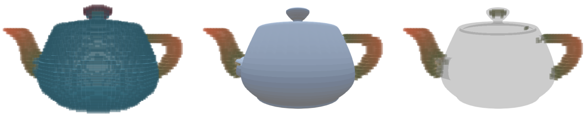


Figure 8: On the left, the union set $A_d \cup B_d$ using the color gradient in terms of the difference set $B_d \setminus A_d$ is visualized. In the center, only the difference set $B_d \setminus A_d$ is drawn on top of the 3d reconstruction. An overlay of the transparent digital twin and the difference set $B_d \setminus A_d$ to identify deviations inside the mesh is shown on the right.

A visualization using the voxel-based representation and a color gradient indicating the strength of the deviation is shown in Figure 8. On the left, the union $A_d \cup B_d$ is shown, which consists of both difference sets $A_d \setminus B_d$, $B_d \setminus A_d$ and the intersection $A_d \cap B_d$. In the middle, the difference set $B_d \setminus A_d$ is rendered on top with the mesh used as digital twin. Here, the voxels are representing the missing spout and handle. An overlay of the difference $B_d \setminus A_d$ on top of the transparent 3d reconstruction is shown on the right to identify deviations of the mesh.

The voxel color gradient from yellow to red indicates the strength of the found deviation and is realized by determining the distances between a dense voxel and the surface of the corresponding mesh, where the deviations are detected. This is accomplished by using the normal of the nearest vertex to that voxel as the direction for a ray starting from the position of the voxel to determine the intersection with the surface and thus the distance between the voxel and the mesh. If the ray does not intersect a surface, the distance between the voxel and the nearest vertex is used instead. This allows using a threshold for the detected deviations to determine at which distance the maximum color intensity for deviations is used (e.g., all deviations greater than 2mm are colored with a maximum color intensity).

The results of this representation based on the 3d reconstruction of the printed teapot using a dense voxel size of 1mm are shown in Figure 9. The reconstructed teapot consists of 97,824 triangles compared to the planning data with only 6,320 triangles. In this example, the deviations detected relatively to the planning data ($B_d \setminus A_d$) are superimposed with the reconstructed mesh (left) and vice versa. The middle contains the intersection set. The objects rendered on the top are using an orthographic projection and the objects at the bottom

	Sparse			Dense		
Run	Resolution	Size	Count	Resolution	Size	Count
1	10x5x10 voxel	1.0cm	500 voxel	5x5x5 voxel	2mm	125 voxel
2	20x10x20 voxel	0.5cm	4.000 voxel	5x5x5 voxel	1mm	125 voxel
3	20x10x20 voxel	0.5cm	4.000 voxel	10x10x10 voxel	0.5mm	1.000 voxel
4	40x20x40 voxel	0.25cm	32.000 voxel	10x10x10 voxel	0.25mm	1.000 voxel
5	40x20x40 voxel	0.25cm	32.000 voxel	20x20x20 voxel	0.125mm	8.000 voxel

Table 1: Resolution, voxel size and voxel count of the sparse and dense voxel grids.

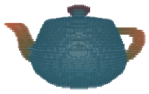
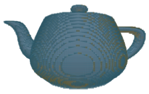
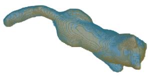
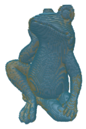
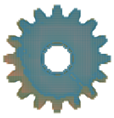
Object	R	$\|A\|$	$\|B\|$	$\|A \cap B\|$	$\|A \setminus B\|$	$\|B \setminus A\|$	$\|A \cup B\|$	DI	JI	KI
 3.0x1.5x1.9cm	1	371	408	355	16	37	408	0.930	0.870	0.931
	2	1,454	1,618	1,430	24	188	1,642	0.931	0.870	0.934
	3	5,794	6,537	5,706	88	831	6,625	0.925	0.861	0.929
	4	23,817	26,842	23,221	596	3,621	27,438	0.917	0.846	0.920
	5	95,800	108,047	92,712	3,088	15,335	111,135	0.909	0.834	0.912
 6.4x3.2x4.0cm	1	1,873	1,787	1,544	329	240	2,113	0.845	0.731	0.845
	2	7,520	7,264	5,701	1,819	1,563	9,083	0.771	0.627	0.771
	3	30,442	29,748	18,314	12,128	11,434	41,876	0.608	0.437	0.608
	4	120,966	119,610	47,401	73,565	72,209	193,175	0.394	0.245	0.394
	5	485,307	481,469	105,613	379,694	375,856	861,163	0.218	0.122	0.218
 9.0x3.2x2.3cm	1	1,653	1,692	1,525	128	167	1,820	0.911	0.838	0.911
	2	6,775	6,883	5,291	1,484	1,592	8,367	0.775	0.632	0.775
	3	27,400	27,371	15,668	11,732	11,703	39,104	0.572	0.401	0.572
	4	110,054	108,950	35,577	74,477	73,373	183,427	0.324	0.193	0.324
	5	440,181	435,901	73,386	366,795	362,515	802,696	0.167	0.091	0.167
 3.8x5.5x3.0cm	1	1,752	1,783	1,604	148	179	1,931	0.907	0.834	0.908
	2	7,059	7,287	5,919	1,140	1,368	8,427	0.825	0.702	0.825
	3	28,353	29,695	19,426	8,927	10,269	38,622	0.669	0.503	0.670
	4	113,593	119,734	48,181	65,412	71,553	185,146	0.413	0.260	0.413
	5	454,733	379,184	99,767	354,966	379,184	833,917	0.213	0.119	0.213
 3.8x0.7x3.8cm	1	558	399	388	170	11	569	0.810	0.682	0.834
	2	2,381	2,628	2,093	288	535	2,916	0.835	0.717	0.837
	3	9,692	13,472	8,715	977	4,757	14,449	0.752	0.603	0.773
	4	38,459	28,276	13,736	24,723	14,540	52,999	0.412	0.259	0.421
	5	154,927	145,666	33,655	121,272	112,011	266,938	0.223	0.126	0.224

Table 2: Results of the shape analysis based on the different voxel sets (hulls) to define the similarity $DI, JI, KI \in [0,1]$.

are rendered with a perspective projection. Figure 10 shows a top and side view of the superimposed difference set $B_d \setminus A_d$ and the reconstructed mesh using the color gradient.

The difference sets $A_d \setminus B_d$ and $B_d \setminus A_d$ resulting at a dense voxel size of 0.25mm using the color gradient are shown in Figure 11. This result represents the fourth run of the second object in Table 3, where $A_d \setminus B_d$ consists of 73,565 voxel and $B_d \setminus A_d$ consists of voxel 72.209. Analyzing both difference sets with the use of the raycast method described in Chapter 5 allows an identification, whether a voxel is located inside or outside of the mesh. This means that the deviation associ-

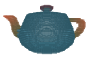
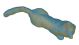
Object	R	|A|	|B|	|A ∩ B|	|A \ B|	|B \ A|	|A ∪ B|	DI	JI	KI
	1	328	397	324	4	73	401	0.893	0.807	0.901
	2	2,570	2,651	2,570	0	81	2,651	0.984	0.969	0.984
	3	19,842	20,375	19,786	56	589	20,431	0.983	0.968	0.984
	4	139,910	144,518	139,398	512	5,120	145,030	0.980	0.961	0.980
	1	3,383	2,960	2,951	432	9	3,392	0.930	0.869	0.934
	2	27,286	24,366	24,306	2,980	60	27,346	0.941	0.888	0.944
	3	217,708	193,982	193,309	24,399	673	218,381	0.939	0.885	0.942
	4	1,741,196	1,553,114	1,547,720	193,476	5,394	1,746,590	0.939	0.886	0.942
	1	1,998	1,900	1,820	178	80	2,078	0.933	0.875	0.934
	2	15,564	15,112	14,518	1,046	594	16,158	0.946	0.898	0.946
	3	124,049	121,030	116,125	7,924	4,905	128,954	0.947	0.901	0.947
	4	992,360	968,189	928,345	64,015	39,844	1,032,204	0.947	0.899	0.947
	1	2,000	1,933	1,885	115	48	2,048	0.958	0.920	0.958
	2	16,356	15,763	15,492	864	271	16,627	0.965	0.931	0.965
	3	130,248	125,852	122,836	7,414	3,016	133,264	0.959	0.921	0.959
	4	1,041,852	1,001,092	982,146	59,706	18,946	1,060,798	0.961	0.925	0.961
	1	369	261	256	113	5	387	0.812	0.684	0.837
	2	2,779	2,660	2,633	146	27	2,806	0.968	0.938	0.968
	3	22,115	21,240	21,006	1,109	234	22,349	0.969	0.939	0.969
	4	178,507	169,520	166,079	12,428	3,441	181.948	0.954	0.912	0.955

Table 3: Results of the shape analysis based on the different voxel sets (volumetric) to define the similarity $DI, JI, KI \in [0,1]$.

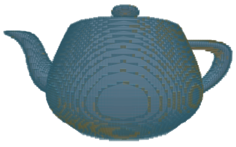
Object	Run	Hull	Volume
	1	00:03	00:02
	2	00:13	00:13
	3	01:02	01:43
	4	04:54	15:38

Table 4: Runtime in mm:ss *format from run 1 to 4 for the voxelization process of the teapot resulting in voxelized hulls or volumes. The results reflect the processing time for the second object of Table 2 (hull) compared to the first object of Table 3 (volume).*

ated to this voxel relates to the inside or outside of the mesh. Figure 12 illustrates the context. In the top row, first the difference set $A_d \setminus B_d$ is visualized. Here, the red voxels are representing outliers while yellow ones are representing inliers. Consequently, this has to be inverted for the differences set $B_d \setminus A_d$, as shown in the top right. The intersection set is shown in the middle. In the bottom row, the dense set A_d (left) and dense set B_d (right) is shown. In addition, the color gradient is used from red to magenta for outliers and orange to yellow for inliers to indicate the strength of the deviation.

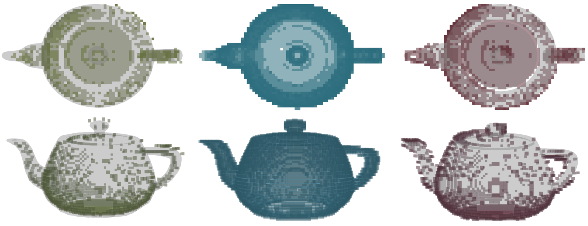


Figure 9: In the top row contains an overlay of the difference set $B_d \setminus A_d$ (yellow), the intersection set $A_d \cap B_d$ (blue) and the difference set $A_d \setminus B_d$ (red) with transparent digital twin from above using a orthographic projection. In the bottom row, the same sets are rendered on top of the transparent 3d object from the side using a perspective projection.

Mesh-based Representation

Transfering detected deviations from the voxelization to its corresponding mesh is done via vertex colors that are associated to the voxel, in which a given vertex is inside. Since the elements of the sparse voxel set $A_s \cap B_s$ are compared in pairs to enable a more efficient shape analysis on the second level, the vertices that are inside in each of these voxel pairs have to be determined. Therefore, it is necessary to walk through all vertices

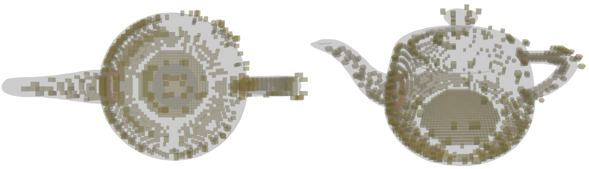


Figure 10: The left side shows a top view of the difference set $B_d \setminus A_d$ on top of the transparent 3d mesh using the color gradient to illustrate the strength of deviations. The screenshot to the right contains a side view of the scene.

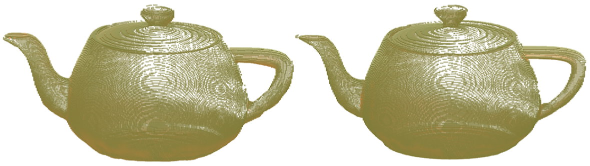


Figure 11: Difference sets $A_d \setminus B_d$ (left) and $B_d \setminus A_d$ (right) resulting at a dense voxel size of 0.25mm (fourth run of the second object in Table 3). In addition, the color gradient is used to illustrate the strength of the deviations.

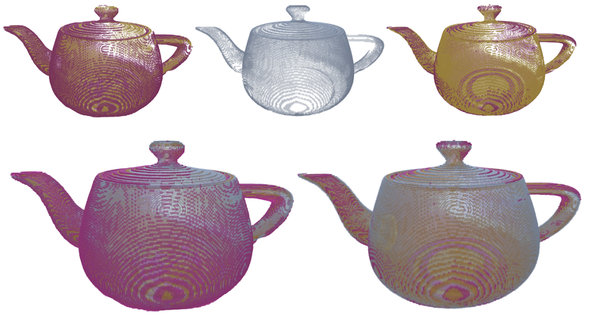


Figure 12: Top row: difference $A_d \setminus B_d$, intersection $A_d \cap B_d$ and difference $B_d \setminus A_d$. Red voxels are representing outliers, yellow voxels are representing inliers. Bottom row: A_d and B_d, outliers and inliers are combined with the color gradient.

of the planning data and 3d reconstructed meshes and check, which vertices are within both sets A_s and B_s.

To reduce the costs, each sparse voxel of sets A_s and B_s has information about the vertices that are associated with a voxel cell and its nearest neighbors (kernel) via the vertex indices. This means, that there has only to be checked, which vertices that are related to this sparse voxel cell and its neighbors intersects with the voxels of the dense voxel sets A_d and B_d. This results in an overall representation of the determined deviations provided by the high-resolution second level voxelization and corresponding shape analysis (see Figure 13).

Real-time Visualization

As already mentioned at the end of the Multi-Level Voxelization Subsection in Chapter 5, the resulting voxel sets are written to files during the voxelization and shape analysis process to reduce memory usage, but also to allow a high resolution voxelization, which is performed step by step using the proposed MLV. Another advantage is that the voxel sets are saved persistently and the shape similarity can be recalculated whenever the data is loaded without a new comparison.

To alleviate performance issues, rendering the high resolution voxel sets in a standard Unity application (where every voxel represents one interactive object), a prototypical loader using Unity DOTS (Data-Oriented Technology Stack) that depends on the ECS (Entity Component System)[5] paradigm [MG20], was also implemented. Compared to a standard Unity application, this results in a significant performance gain, as shown in Table 5.

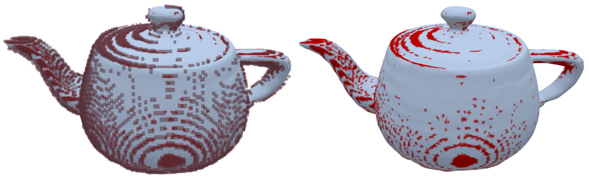


Figure 13: Left: combined rendering of the difference set $A_d \setminus B_d$ and the reconstructed teapot mesh. Right: determined deviations transferred to mesh data as final visualization for fast rendering and easy data sharing.

DS	|A| + |B|	*VoxMesh*	*VoxViz*
2mm	3,345	∼355.6 fps	∼744.8 fps
1mm	13,658	∼119.5 fps	∼504.4 fps
0.5mm	54,771	∼21.7 fps	∼211.6 fps
0.25mm	219,004	∼4.6 fps	∼65.1 fps
0.125mm	876,082	∼1.0 fps	∼16.3 fps

Table 5: Comparison of our VoxMesh and VoxViz tools for real-time rendering of the cat mesh's voxel sets.

7 RESULTS AND DISCUSSION

In addition to the resolution of the voxelization and the vertex count of the geometries to be compared, the performance also depends on the selected types of result presentation. The latter consist of representing the deviations either voxel-based using only one color for each resulting voxel set (see Figure 9) or two colors for each difference set determining inliers or outliers with the option, to also use a gradient to visualize the strength of the deviations (see Figure 10, 12), or a mesh-based representation, where deviations are visualized by coloring the mesh at the corresponding locations (see Figure 13). For example, the fourth run of the second object (reconstructed teapot) in Table 2 using the setup mentioned at the end of the Results and Evaluation Subsec-

[5] `https://docs.unity3d.com/Packages/com.unity.entities@0.51/manual/ecs_core.html`

tion in Chapter 5 took 04:54 min using one color for each voxel set, 10:20 min with a enabled gradient and 07:42 min using the mesh-based representation. This is due to the fact that voxel-based representations using the gradient and mesh-based representations are considering the vertices of the meshes for visualization.

Concerning shape analysis and the compared statistical methods (DI, JI, KI as defined in Section 5) it can be mentioned, that the results of the DI and KI are mostly similar, which means that a choice between them using as a threshold does only matter if the result has to be very precise. In terms of JI, the results are a little bit lower compared to DI and KI.

Regarding the shape analysis between the voxelized hulls and interior volumes (see Table 2 and Table 3), it can be stated that this results in a higher deviation or lower shape similarity with decreasing voxel size using the voxelized hulls. On the one hand, this is in close correlation with the previous alignment and geometric registration of the two geometries to be compared, which also had to be corrected afterwards using a PCA.

On the other hand, the shape analysis is affected by the uneven surface structures of flat surfaces regarding the reconstructed meshes, if a high resolution (e.g., voxel size < 0.5mm) is chosen. With respect to the volumetric voxel sets, the resolution of the voxel grids did not have a large effect on the shape similarity results, as the resulting voxel sets only grew strongly in terms of intersection, which was significantly higher compared to the intersection of voxelized hulls with the same grid settings.

Errors or deviations that can occur during the scanning process and the 3d mesh reconstruction are not taken into account. In terms of scanning, the occurance of errors is strongly correlated with the used scanning devices (in our case out-of-the-shelf hardware), scan settings and acqusition conditions (e.g., print material and lighting). For the mesh reconstruction the included software (ExScan) without any mesh optimizations was used to avoid deformations based on that opmizations.

The determined deviations are either visualized using a voxel-based representation or mesh-based representation. Using the voxel-based representation allows to visualize missing parts by using the voxelized geometry from the counterpart mesh, where this parts exists (see Figure 7). In addition, a threshold can be used for a maximum permitted deviation. All deviations smaller than this threshold are colored using a gradient based on the distance of the voxel to the next vertex in the mesh. All deviations above (e.g., deviation > 1mm) are colored using the maximum gradient.

Another advantage is based on the overlay of the voxel sets and the transparent mesh, which makes it possible to identify whether an individual voxel is outside or inside the object (see Figures 8 and 10). In addition, inliers and outliers can also be classified during the MLV (see Figure 12). Regarding missing parts using our implementation of the mesh-based representation, a visualization of missing parts is only possible for objects, where this parts exist, by coloring these mesh parts (see Figure 13).

Our prototypical implementation also allows saving the center positions of all resulting voxel sets that can be either used as point cloud input or for reproducing the voxel sets in other animation software such as Blender with the use of geometry nodes.[6] In addition, the resulting voxel sets can also be persistently saved and loaded using an internal format, but also using the *.obj* format, to support a wide range of third party 3d tools.

8 CONCLUSION & FUTURE WORK

In this paper, a concept for visualizing deviations between topologically different 3d planning data and the corresponding 3d reconstruction of 3d printed objects based on a multi-level voxelization was presented and prototypically implemented as a Unity application. During the voxelization process, a shape analysis using statistical methods was performed on the voxelization result to determine the shape similarity of the planning data along with its 3d reconstruction. Furthermore, this included a comparison between the resulting voxel sets, which either consisted of the voxelized hulls of the meshes or volumetric data. Determined deviations are either visualized using a voxel- or mesh-based representation to easily find errors in the 3d printed object.

In this context, further work regarding the voxelization is subject to continuous development and optimization, which also involves the initial overlay and alignment. This could be extended, for example, by using the ICP (Iterative Closest Point) technique. With regards to the voxelization of the meshes, the process could also be implemented using Unity's ECS system, as it is already used in the context of the result visualization in our VoxViz tool, which in addition to the associated increase in performance additionally enables a parallelization of the workflows (like parallelized comparison of several voxels of the sparse grid).

For the voxel-based representation a gradient was used to visualize the strength of found deviations, which has not been applied yet to the mesh-based representation to colorize the mesh surface depending on the strength of the deviations. Since a visualization of the voxelization result in the 1 to 2 millimeter range can also be performed by mobile devices, an overlay of the voxelized surfaces with the printed object is also conceivable in order to illustrate deviations using augmented reality.

[6] `https://docs.blender.org/manual/en/latest/modeling/geometry_nodes/index.html`

9 REFERENCES

[BSSW21] Sebastian Bickel, Christopher Sauer, Benjamin Schleich, and Sandro Wartzack. Comparing cad part models for geometrical similarity: A concept using machine learning algorithms. *Procedia CIRP*, 96:133–138, 2021. 8th CIRP Global Web Conf. - Flexible Mass Customisation (CIRPe 2020).

[CHL+17] Xin Chen, Jingbin Hao, Hao Liu, Zhengtong Han, and Shengping Ye. Research on similarity measurements of 3d models based on skeleton trees. *Computers*, 6:17, 04 2017.

[DFFW18] Jozef Doboš, Carmen Fan, Sebastian Friston, and Charence Wong. Screen space 3d diff: a fast and reliable method for real-time 3d differencing on the web. In *Proc. Web3D '18*. ACM, 2018.

[DGJ20] Andreas Dietze, Paul Grimm, and Yvonne Jung. Visualization of differences between spatial measurements and 3d planning data. In *In Proceedings of the 25th Intl. Conf. on 3D Web Technology*, pages 1–5. ACM, 2020.

[DGJ21] Andreas Dietze, Paul Grimm, and Yvonne Jung. Updating 3d planning data based on detected differences between real and planning data of building interiors. In Vaclav Skala, editor, *29th Intl. Conf. on Computer Graphics, Visualization and Computer Vision (WSCG '21)*, pages 1–6, Plzen, Czech Republic, 2021. CSRN.

[FI18] Sam Fletcher and Md Islam. Comparing sets of patterns with the jaccard index. *Australasian Journal of Information Systems*, 22, 03 2018.

[KWL20] Ha-Seong Kim and Myeong Won Lee. 3d object recognition using x3d and deep learning. In *25th Intl. Conf. on 3D Web Technology*, Web3D '20, New York, NY, USA, 2020. ACM.

[LEAM+19] Zouhir Lakhili, Abdelmajid El Alami, Abderrahim Mesbah, Aissam Berrahou, and Hassan Qjidaa. 3d shape classification using 3d discrete moments and deep neural networks. In *Proc. of 2nd Intl. Conf. on Networking, Information Systems and Security*, NISS19, New York, NY, USA, 2019. ACM.

[MG20] Mathieu Muratet and Délia Garbarini. Accessibility and serious games: What about Entity-Component-System software architecture? In *GALA 2020*, Laval, France, December 2020.

[MR01] Novotni Marcin and Klein Reinhard. A geometric approach to 3d object comparison. In *Proceedings Intl. Conf. on Shape Modeling and Applications*, pages 167–175, 2001.

[NZL+20] Weizhi Nie, Yue Zhao, An-An Liu, Zan Gao, and Yuting Su. *Multi-Graph Convolutional Network for Unsupervised 3D Shape Retrieval*, pages 3395–3403. ACM, New York, NY, USA, 2020.

[OGBS06] Ipek Oguz, Guido Gerig, Sebastien Barre, and Martin Styner. Kwmeshvisu: A mesh visualization tool for shape analysis. *The Insight Journal*, 2006.

[RMWQ19] Baltej Singh Rupal, Khaled G. Mostafa, Yeping Wang, and Ahmed Jawad Qureshi. A reverse cad approach for estimating geometric and mechanical behavior of fdm printed parts. *Procedia Manufacturing*, 34:535–544, 2019. 47th SME North American Manufacturing Research Conf., NAMRC 47, Pennsylvania, USA.

[WWWH21] Martin Weinmann, Sven Wursthorn, Michael Weinmann, and Patrick Hübner. Efficient 3d mapping and modelling of indoor scenes with the microsoft hololens: A survey. *PFG - Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 89:319 – 333, 2021.

[WZL+20] Bing Wang, Shuncong Zhong, Tung-Lik Lee, Kevin S Fancey, and Jiawei Mi. Non-destructive testing and evaluation of composite materials/structures: A state-of-the-art review. *Advances in Mechanical Engineering*, 12(4):1687814020913761, 2020.

[ZAB+16] Fatima Rafii Zakani, Khadija Arhid, Mohcine Bouksim, Taoufiq Gadi, and Mohamed Aboulfatah. Kulczynski similarity index for objective evaluation of mesh segmentation algorithms. In *5th Intl. Conf. on Multimedia Computing and Systems (ICMCS)*, pages 12–17, 2016.

[ZWB+04] Kelly Zou, Simon Warfield, Aditya Bharatha, Clare Tempany, Michael Kaus, Steven Haker, William Wells, Ferenc Jolesz, and Ron Kikinis. Statistical validation of image segmentation quality based on a spatial overlap index. *Academic radiology*, 11:178–89, 02 2004.