An Augmented Reality App supporting Service Technicians based on 3D Reconstruction and Object Recognition

Marian Bookhahn[1], Felix Essig[2], Tara Lorenz[3], Florian Voit[2], Frank Neumann[1]

[1]HTW Berlin, [2]CADENAS GmbH, [3]GFaI e.V.

E-Mails: marian.bookhahn@htw-berlin.de, f.essig@cadenas.de, lorenz@gfai.de, f.voit@cadenas.de, frank.neumann@htw-berlin.de

Abstract: The presented augmented reality app supports service technicians in scenarios, where the article number of an urgently needed spare part cannot be determined ad-hoc because the system documentation is either not available or not up to date. For this purpose, the app offers means for the identification and ordering of suitable spare parts based on a 3D scan of the visible and preserved component geometry. For this purpose, the project opted to use the recently released LiDAR sensor of the iPad Pro and iPhone Pro to provide the input data of the 3D reconstruction and recognition pipeline. Apple's ARKit supplies the input depth images, the trajectories and pose data. Within this pipeline, a TSDF based fusion approach is employed for 3D reconstruction. Regarding segmentation, extensive experiments were conducted to identify segmentation methods offering both reliable results and sufficient performance. For object recognition different approaches were developed that can cope with incomplete model data. Within the app's user interface augmented reality features are used to blend data from different sources. When conducting experiments especially in facility management, the proposed approach shows promising results.

Keywords: Augmented Reality, ARKit, 3D Reconstruction, TSDF, Object Recognition

1 Introduction

Service technicians often encounter problems if they repair e.g., production systems, escalators or elevators and cannot reliably determine the article number of an urgently needed spare part on site. A 3D scan with a mobile device and the interactive selection of the part, they are looking for, can significantly speed up the search based on an online database. This is where the *SparePartAssist* project comes in: The application enables the service technician to quickly identify the required spare part. To do this, she uses her smartphone to perform a 3D scan of the component to be replaced. Since components are often not easy to recognize because they are either installed and not completely visible, heavily worn or partially destroyed, the search algorithm can also recognize components based on fragments of the geometry. Based on the article data determined in this way, the service technician can then order the right spare part.

In this paper, we make the following contributions:

- Concept, implementation, and testing of a pipeline using depth images of the LiDAR sensor as input data for subsequent 3D reconstruction, segmentation, and object recognition. A fusion approach based on a Truncated Signed Distance Function (TSDF) has been adopted for 3D reconstruction, which has been further enhanced to deal with different voxel sizes depending on the object's distance. Regarding segmentation, extensive experiments were conducted to identify segmentation methods offering both reliable results and sufficient performance. For object recognition, different approaches were developed that can cope with incomplete model data.
- Augmented reality (AR) features are discussed and employed for different parts of the app's user interface. Here we benefit from the fact that AR allows to fuse image data from the real scene with information gathered by the different constituents of the reconstruction and recognition pipeline.

This paper is organised as follows: The next section discusses related research. In section 3 a set of requirements for the app will be laid out. Section 4 describes the different aspects of the app's concepts regarding the selected hardware, the employed workflow, the adopted software architecture, the approaches used for 3D reconstruction, segmentation, and object recognition. Subsequently, section 5 discusses aspects of the AR-based user interface. The last section summarises the present research and points out directions for future work.

2 Related Work

For the realisation of the above-mentioned application scenario, methods for object recognition and features of AR frameworks will be discussed in the following.

2.1 AR Frameworks

Augmented Reality supports the real-time blending of a scene with virtually generated objects, markers, and interaction possibilities. For this purpose, various sensors (e.g. RGB camera, depth sensor, inertial measurement unit) deliver the input data for the tracking of the position of the device, which is realised by simultaneous localization and mapping (SLAM) algorithms. These features are typically provided by AR frameworks (e.g. ARKit, ARCore and Vuforia) that comprise a dedicated software development kit (SDK).

AR frameworks may optionally implement the recognition of 2D and 3D objects based on two separate approaches [1]:

- Recognition of 2D objects: image recognition, available in ARKit, ARCore and Vuforia
- Recognition of 3D objects: object recognition, available in ARKit and Vuforia

Unfortunately, the features for 3D object recognition provided by AR frameworks are quite limited [1]: Currently they handle only predefined set of components (in Vuforia only for 10... 20 components). The recognition of 3D objects in ARKit is limited to well-textured objects only. Therefore, such methods supplied by AR frameworks cannot be used for the present use case.

2.2 Image-based Object Recognition

Image-based methods for object recognition have developed to maturity over the last decades. Here, convolutional neural networks (CNN) architectures have permitted huge advances regarding accuracy and speed [2]. Surely, such methods may serve as basis for object recognition in the present application scenario.

2.3 3D Object Recognition

3D object recognition focuses on discovering known 3D models within point clouds from LiDAR systems, laser scanners, stereo cameras, or time-of-flight (ToF). More formally the task of 3D object recognition can be described as follows [3]:

For a given set of 3D models M and a scene S (point cloud), pairs of 3D models M_{kj} and transformations T_j are searched for, which meet the following conditions:

By means of the transformation T_j, the 3D model M_{kj} can be registered in the scene.

Depending on the number of 3D objects in quantity M, different approaches can be distinguished [2]:

1. For **smaller quantities of 3D objects** combinations of geometric hashing and the RANSAC algorithm may be applied [3] at the cost of linear algorithmic complexity with regards to the number of models.
2. 3D point cloud descriptors [4] [5] may serve for the detection of 3D objects within a **larger set**. Here, following types of 3D descriptors may be distinguished [5]:
    a) Local 3D descriptors taking into account the local geometric information of a feature point (e.g. surface normal and curvature)
    b) Global 3D descriptors considering the geometric information of an object's entire point cloud
    c) Hybrid methods combining different 3D descriptors

In principle, these 3D descriptor-based methods may support the described application scenario.

3 Requirements Analysis

In the following, the requirements gathered for the AR app supporting service technicians will be laid out. Starting from these requirements, the concept regarding recognition methods, hardware and software architecture will be developed in the next section. Table 1 lists the identified functional, non-functional, and technical requirements for the AR app.

Table 1: Identified functional, non-functional, and technical requirements for the AR app

ID	Group	Description
F-1	Object recognition of spare parts by service technicians	
F-1.1		Object recognition for typical components in the field of mechanical engineering, plant engineering and facility management.
F-1.2		Object recognition for components that are either installed and not completely visible, heavily worn or partially destroyed.
F-1.3		Object recognition for textured and non-textured parts.
F-1.4		Object recognition under a wide range of light conditions. .
F-2	Component catalogues	Component data is collected in very large catalogues comprising up to 100.000 items.
NF-1	Recognition time	The time required to identify a needed spare part should not exceed 2 minutes.
T-1	Devices	Object recognition should be conducted using mobile devices (e.g. smartphones and tablets).

4 Concept

The employed object recognition method should be able to identify individual instances of spare parts that belong to semantic categories e.g., motors, controller, valves, pumps. As stated in requirement F-1.2, a partial view of the item should be sufficient for the identification of the needed spart part. Consequently, 3D descriptor-based object recognition methods supporting partial views were adopted.

In the following, key concepts of *SparePartAssist* regarding the selected hardware, the employed workflow, the adopted software architecture, the approaches used for 3D reconstruction, segmentation, and object recognition will be described.

4.1 Hardware

According to the selected 3D object recognition approach, mobile devices equipped with depth

sensors will be considered as hardware platform for *SparePartAssist*. At the very beginning of the project, we analysed different types of depth sensors currently available in mobile device [6] [7] [2]. Unfortunately, the data quality of iToF sensors used in Android devices suffers from multiphase interference issues. Therefore, we opted for dToF sensors combined with an illumination by a scanning laser beam as employed in the LiDAR sensor of the iPad Pro and iPhone Pro. Table 2 gives an overview on the characteristics of this sensor:

Table 2: Data of the iPad Pro and iPhone Pro LiDAR sensor [2]

Model	Apple LiDAR (iPad Pro 2020)
Resolution	256 x 192
Frame rate	60 FPS
Horizontal field of view	~ 62 deg
Vertical field of view	~ 48.5 deg

4.2 Workflow

Various sensors (RGB camera, LiDAR sensor, IMU) of the iPad Pro and iPhone Pro deliver the input for the 3D reconstruction and recognition pipeline. In a first step, the SLAM algorithm used by Apple's ARKit calculates the camera pose for each camera frame based on the provided sensor data (cf. Figure 1).

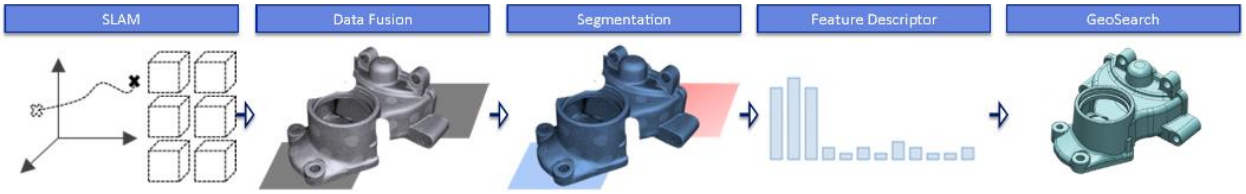

Figure 1: Workflow of *SparePartAssist* [8]

The goal of the second step is to create a mesh representation based on the depth images and the associated camera poses. For this purpose, the dedicated z-Fusion library employs a TSDF-based fusion algorithm that accumulates the depth images since the start of the scan process. In a third step, the user selects within the generated mesh the region containing the spare part by means of a bounding box. The parts of the mesh laying within the bounding box will then be segmented. For the fourth step, the user choses coloured patches of the segmented mesh belonging to the spare part. For the selected patches of the mesh, a feature descriptor will be calculated next. This feature descriptor is used as input for the fifth step of the workflow, where the *GeoSearch* determines geometrically similar components discovered in the catalogue. Finally, a list of spare parts in rank of geometrical similarity is presented to the service technician as a proposal to choose from.

4.3 Software Architecture

Apache Cordova [9] was selected as overall framework for the app, which allows using a platform neutral HTML and JavaScript based user interface and app that may access platform specific code through dedicated plugins. As the dToF sensor is currently available on iOS only, Apache Cordova supports on the one hand the present implementation of the app on iOS. On the other hand, it provides an option for a future implementation on Android, once dToF sensors might become available on this platform as well.

The Swift-based data acquisition plugin constitutes the central component of the software architecture (cf. Figure 2) as it makes the various sensor data provided by ARKit available to the user interface. The user interface makes use of *React*, *TypeScript* and *ThreeJS* for 3D visualisation.

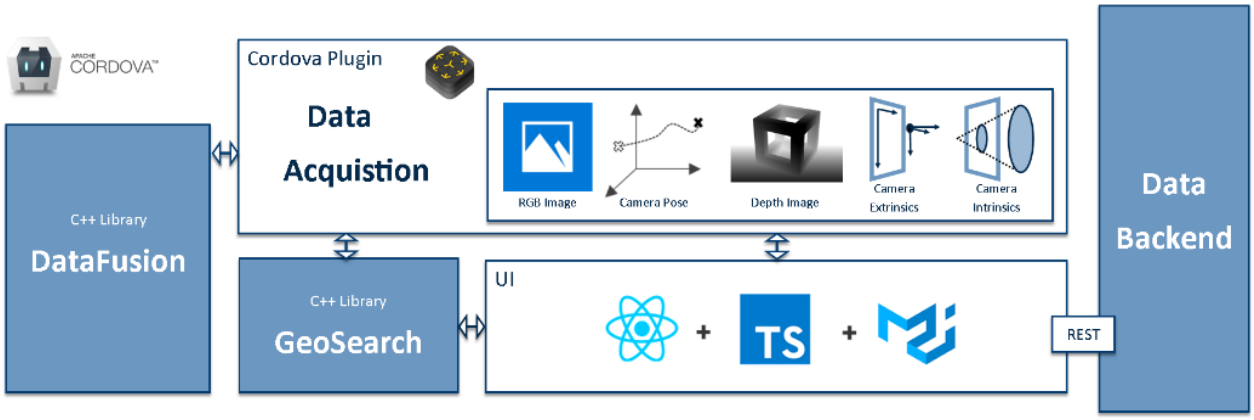

Figure 2: App architecture based on Apache Cordova [2]

The C++ based z-Fusion library (indicated in Figure 2 as data fusion) has been made available to the data acquisition plugin through a C wrapper bridge. The z-Fusion library receives depth images, camera poses and camera characteristics from the data acquisition and sends fused mesh data in return.

For the sake of simplicity during the development phase, the feature descriptor is calculated by the RESTful *GeoSearch* webservice at the backend. Once the feature descriptor has been established and validated, its calculation may be moved to the app side as indicated by the C++ library *GeoSearch* in Figure 2.

4.4 3D Reconstruction

The 3D reconstruction is organised in two stages: The first stage fuses the depth images along with their pose information into a TSDF representation. As mobile devices are memory constrained, a sparse structure was chosen, such that any observed surface area will correspond to a bounded (thin) volume. The total volume is subdivided into blocks of a fixed size, where size means the number of grid points in a classical TSDF representation. Those blocks are organized into a hash map and addressed by their truncated integral coordinates in a suitably chosen coordinate system,

independent from the coordinate system used for the pose.

The second stage turns the information within the TSDF representation into a triangle mesh. To this means, a version of the marching cubes algorithm using the base cases of Lengyel [10] gets applied. Special care was taken to ensure that the order of operations while computing the vertices are independent of the arrangement of a point-pair within a grid cube. As a result, no holes or cracks will be introduced by numerical instabilities.

These two stages are fast enough to run in real-time, at a rate of at least 1 Hz on the target hardware, when the maximum depth is cut at 1.3 meters and an appropriately chosen grid resolution of 5 mm.

4.5 Segmentation

In order to get partial features from the reconstructed 3D data, automatic segmentation methods are applied to the z-Fusion meshes. Reliable segmentation results due to prominent features were found by implementing a region-growing approach [11]. The mesh surface is auto decomposed into mesh segments where input parameters can be customised to be suitable for scanned mesh data.

Since segmentation of complex meshes requires a rather high amount of computation power, the calculation is performed on an app server: after preview of the scanned mesh on the mobile device the user can start the mesh segmentation. This will send a POST request to the segmentation service where settings as well as the compressed 3D data are transferred.

To receive the segmentation response within a reasonable amount of time some pre-processing of the mesh is required. This is realised by automatic cleaning and customised quadric mesh simplification filters which both improve segmentation performance and stability [12] [13] [14]. Several filter settings have been investigated for rendering well-segmented objects. The server response contains color-coded mesh segments that again can be displayed inside the app preview for selection of the parts of interest.

This approach allows the user to select whole as well as partially hidden objects for geometry search. Although typical scanned objects are not complete due to limited recording time or resolution deficiencies, the most relevant and best retrieved shapes of the object can be selected for fitting the corresponding shapes to the search database.

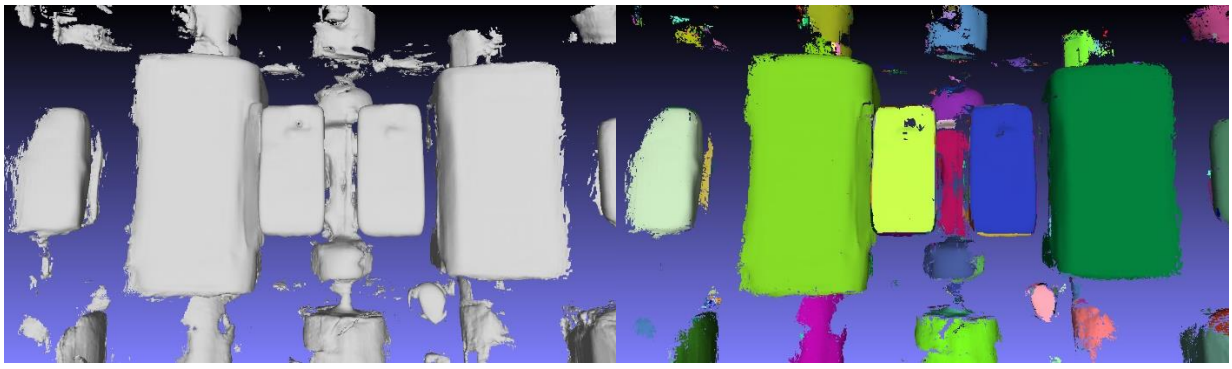

Figure 3: Scanned mesh of heating pumps and panels (*left*); Filtered and segmented mesh (*right*).

4.6 Object Recognition

Important steps towards object recognition for technical applications are primitive fitting and descriptor-based shape retrieval [15]. The former method tries to fit shapes, mostly simple geometric primitives, to the scanned 3D data to gather information about sizes, orientations, and types of shapes [16]. On the other hand, the more abstract descriptor method is based on statistical or transformed geodata of a whole object or a certain part of it [17]. These descriptors can act as fingerprints and be matched to a descriptors database of index parts.

The currently implemented search service benefits from both approaches. When a search part is sent to the search service via POST request the scan must be segmented properly beforehand to match a primitive type as good as possible. Here the app's preview feature allows interactive selection of parts of different levels of detail. Then first a quadric fitting method is used to extract geometrical data from the search part. Secondly, several descriptors are calculated by means of the triangulated data of this partial object.

A search algorithm compares the requested part descriptors to similar descriptors based on different metrics and error tolerances. For now, robust configuration of these tolerances in case of scanned meshes is being investigated. If an indexed part is found, the geometrical data gathered by the fitting method is compared to the corresponding data for this part. The index database can be extended by additional part catalogues or search categories.

5 Augmented Reality based User Interface

Overall, the user interaction design for the app plays a crucial role in the app's development process as service technicians without any specialised IT background form the typical user group. Therefore, the user interaction design was conducted thoroughly. Each different versions of the implemented UI went through a dedicated user testing providing direct input for the next iteration. Overall, the UI consists of several screens guiding the user step-by-step through the search process. The workflow of the service technician starts at the first screen showing previously identified spare parts (cf. Figure 4 on the left). During the next step the user scans visible parts of the required spare part by means of the LiDAR sensor as indicated in Figure 6 on the right. After scanning the part, the user selects the interesting portions of the scan by means of a bounding box, which can be moved, scaled, and rotated (cf. Figure 4 on the right). Here, the portion of the scan contained within the bounding box is highlighted in yellow.

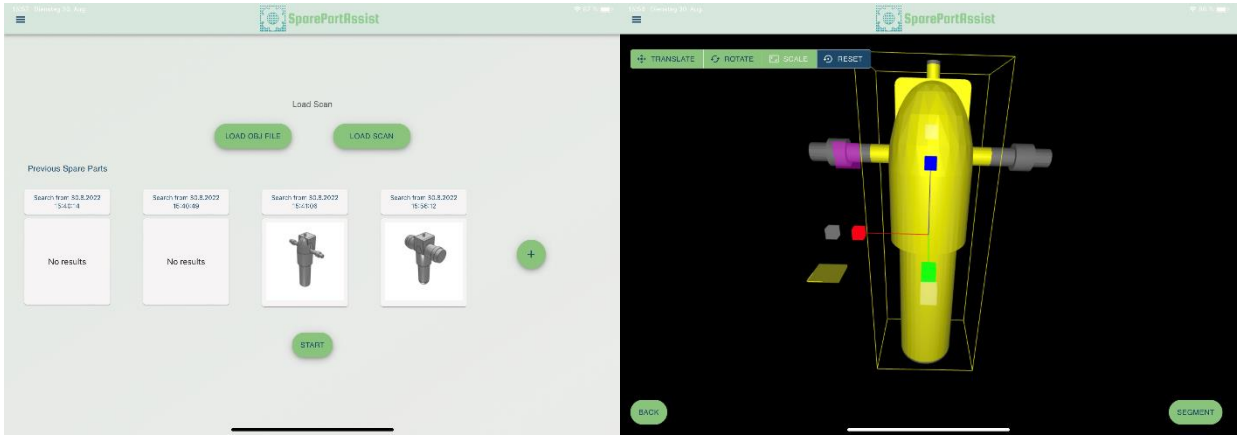

Figure 4: Start screen of the app (*left*); selection of interesting portion through a bounding box (*right*).

Subsequently, the obtained segmentation response is visualised, where different colours indicate different patches of the mesh (cf. Figure 5 on the left). Based on this visualisation, the user selects the segments belonging to the required spare part. For those segments, the object recognition process returns a list of the most similar spare parts contained in the catalogue ranked by similarity values (cf. Figure 5 on the right).

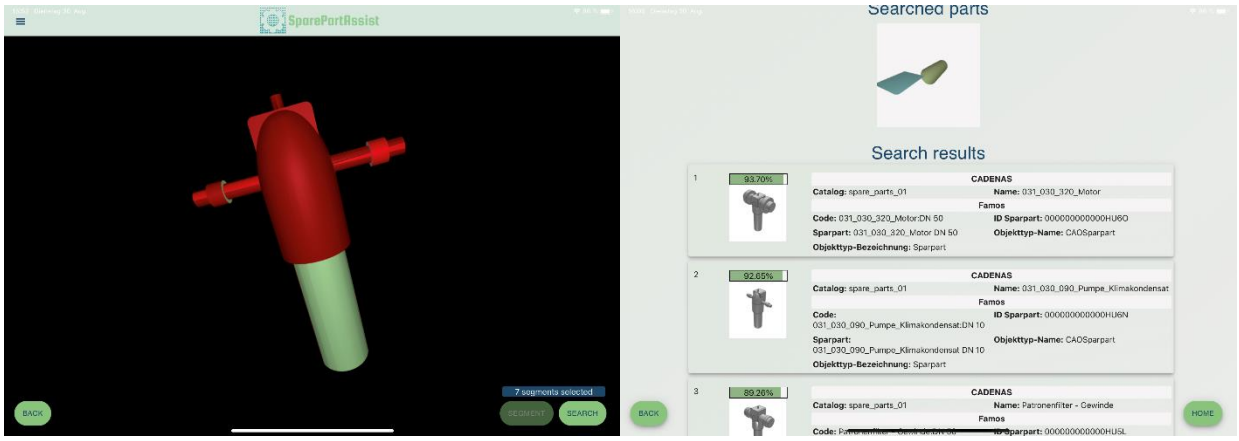

Figure 5: Visualisation and selection of segmentation results (*left*); final screen showing the identified spare parts as search results (*right*).

In the context of the design process, AR was considered as a powerful means to provide the service technicians with additional information calculated by the app, which should be blended with the real scene data.

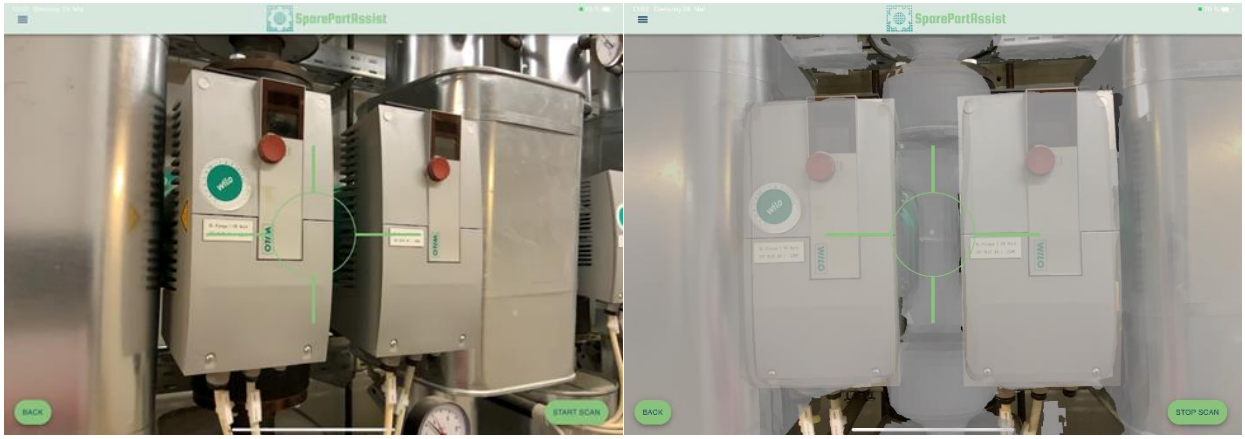

Figure 6: Photo of heating pumps and panels (*left*); AR overlay with preview scanned mesh (*right*).

One prominent example is the page shown during the scan process (cf. Figure 6 on the right), where the mesh calculated by the z-Fusion library is blended on top of the camera data as a transparent overlay. This way the user is guided through the whole scanning process and sees the progress of the mesh creation as a preview to ensure best scan quality. The service technician can easily go back to views without sufficient mesh coverage, which will ultimately contribute new depth frames to the data fusion process and improve the preview mesh.

Similar AR approaches will be adopted for topics such as the integration of assembly and disassembly instructions in the app.

6 Conclusions and Future Work

At the time of writing the complete workflow of the *SparePartAssist* app has been implemented. Service technicians can send scans of a desired spare part to the backend. Currently, the robust configuration of the 3D object recognition algorithm with regards to error tolerances is under way. Once this fine-tuning step is completed, a large validation phase will be conducted in a real world setting in facility management.

Afterwards, subsequent steps will advance the app towards the following objectives:

- Further improvements of the z-Fusion library to deal with different voxel sizes depending on the object's distance in order to handle smaller spare parts
- Integration of the FAMOS facility management system

Acknowledgements

This research has been funded by the Federal Ministry of Education and Research (BMBF) of Germany in the framework of SME Innovative (German: KMU-innovativ) (project number 16SV8511).

References

[1] F. Neumann and T. Kuzyna, “Objekterkennung in Augmented Reality Frameworks. Möglichkeiten und Grenzen von AR-Technologien bei der Suche nach dem passenden Ersatzteil im Außeneinsatz,” in *HTW Berlin Wissenschaftssymposium Grenzen*, Berlin, 2019.

[2] M. Bookhahn, A. Brechtel, T. Lorenz, F. Voit and F. Neumann, "SparePartAssist -a mobile app to identify spare parts based on 3D sensor data. An interim balance," in *3D NordOst 2021*, Berlin, 2021.

[3] C. Papazov and D. Burschka, “An Effcient RANSAC for 3D Object Recognition in Noisy and Occluded Scenes,” in *Proceedings of the 10th Asian Conference on Computer Vision (ACCV'10)*, Queenstown, New Zealand, 2010.

[4] E. J. Jasphin and C. S. Joice, “3D Feature Descriptors and Image Reconstruction Techniques: A Review,” *International Journal of Pure and Applied Mathematics,* pp. 1117-1125, 10 2018.

[5] X.-F. Han, J. S. Jin, J. Xie, M.-J. Wang and W. Jiang, “A comprehensive review of 3D point cloud descriptors,” *ArXiv,* 8 2 2018.

[6] F. Neumann, “SparePartAssist – Welcher Smartphone-Tiefensensor liefert die passenden Punktwolken? – Teil 1 (Android),” 12 01 2021. [Online]. Available: https://sparepartassist.f2.htw-berlin.de/news/21.

[7] F. Neumann and T. Lorenz, “SparePartAssist – Welcher Smartphone-Tiefensensor liefert die passenden Punktwolken? – Teil 2 (iOS),” 01 03 2021. [Online]. Available: https://sparepartassist.f2.htw-berlin.de/news/27.

[8] M. Bookhahn, “SparePartAssist - Architekturposter,” HTW Berlin, Berlin, 2020.

[9] The Apache Software Foundation, “Apache Cordova,” [Online]. Available: https://cordova.apache.org/. [Accessed 30 5 2022].

[10] E. Lengyel, “Transition Cells for Dynamic Multiresolution Marching Cubes,” *Journal of Graphics, GPU, and Game Tools,* vol. 15, no. 2, pp. 99-122, 2010.

[11] A. Andreadis, P. Mavridis and G. Papaioannou, “Facet Extraction and Classification for the Reassembly of Fractured 3D Objects,” in *Eurographics (Posters)*, 2014.

[12] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli and G. Ranzuglia, “MeshLab: an Open-Source Mesh Processing Tool,” in *Sixth Eurographics Italian Chapter Conference*, 2008.

[13] M. Campen, M. Attene and L. Kobbelt, “A Practical Guide to Polygon Mesh Repairing,” in *Eurographics (Tutorials),* 2012.

[14] S. Forstmann and C. Rorden, “Fast quadric mesh simplification,” [Online] https://github.com/sp4cerat/Fast-Quadric-Mesh-Simplification, 2016.

[15] P. Theologou, I. Pratikakis and T. Theoharis, “A review on 3D object retrieval methodologies using a part-based representation,” *Computer-Aided Design and Applications,* vol. 11, no. 6, pp. 670--684, 2014.

[16] J. Andrews and C. H. Séquin, “Type-constrained direct fitting of quadric surfaces,” *Computer-Aided Design and Applications,* vol. 11, no. 1, pp. 107--119, 2014.

[17] B. Bustos, D. A. Keim, D. Saupe, T. Schreck and D. V. Vranić, “Feature-based similarity search in 3D object databases,” *ACM Computing Surveys,* vol. 37, no. 4, pp. 345--387, 2005.