Proceedings of Meetings on Acoustics

Volume 44 http://acousticalsociety.org/

6th Underwater Acoustics Conference & Exhibition

20-25 June 2021

Signal Processing in Acoustics: Underwater Unexploded Ordnance (UXO) Detection and Remediation

3D rendering of shipwrecks from synthetic aperture sonar

Ole Jacob Lorentzen
Department of Informatics, University of Oslo: Universitetet i Oslo, OSLO, NO-0316, NORWAY; olejlor@ifi.uio.no

Roy Edgar Hansen, Torstein Olsmo Sæbø, Stig Asle Vaksvik Synnes and Marc Geilhufe
Norwegian Defence Research Establishment: Forsvarets forskningsinstitutt Kjeller, NORWAY; roy-edgar.hansen@ffi.no; t orstein-olsmo.sabo@ffi.no; stig-asle.synnes@ffi.no; marc.geilhufe@ffi.no

Synthetic aperture sonar (SAS) bathymetric mapping provides centimeter-scale horizontal resolution of the seafloor. Typically, SAS data has been visualized as 2D images or gridded 2.5D surfaces, but it can also be plotted as 3D point clouds. Even if the imaging system is not fully volumetric, the imaging geometry leaves the possibility of having data points that reside in the same lateral positions in a ground coordinate system. Visualizing these without geometric distortion or loss of data is challenging. In this paper we investigate the use of full-fledged 3D rendering with artificial lighting and shading techniques in order to convey more of the available information. We aim to produce better 3D visualization of shipwrecks. We provide an overview of which data products are available from state-of-the-art SAS processing, and explore existing visualization methods. We discuss the use of surfaces and point clouds for 3D rendering, and various shading and rendering techniques. Our goal is to display SAS bathymetry in a way that is both intuitive and can convey more supplemental information to the observer. Finally, we show image and video examples of rendering large shipwreck scenes using data collected by a HISAS1032 SAS carried by a HUGIN autonomous underwater vehicle.

Published by the Acoustical Society of America

 https://doi.org/10.1121/2.0001483

1. SYNTHETIC APERTURE SONAR

High resolution synthetic aperture sonar (SAS) is a signal processing technique which increases the along-track resolution in sonar images by coherent combination of collected pings.[1] By interferometric processing, a high resolution depth map (bathymetry) can also be produced.[2] The HISAS 1032 SAS system used in this paper provides resolution cells of approximately 2 cm in both horizontal dimensions for SAS backscatter images, and resolution cells of about 20 cm in both horizontal dimensions in the depth map.

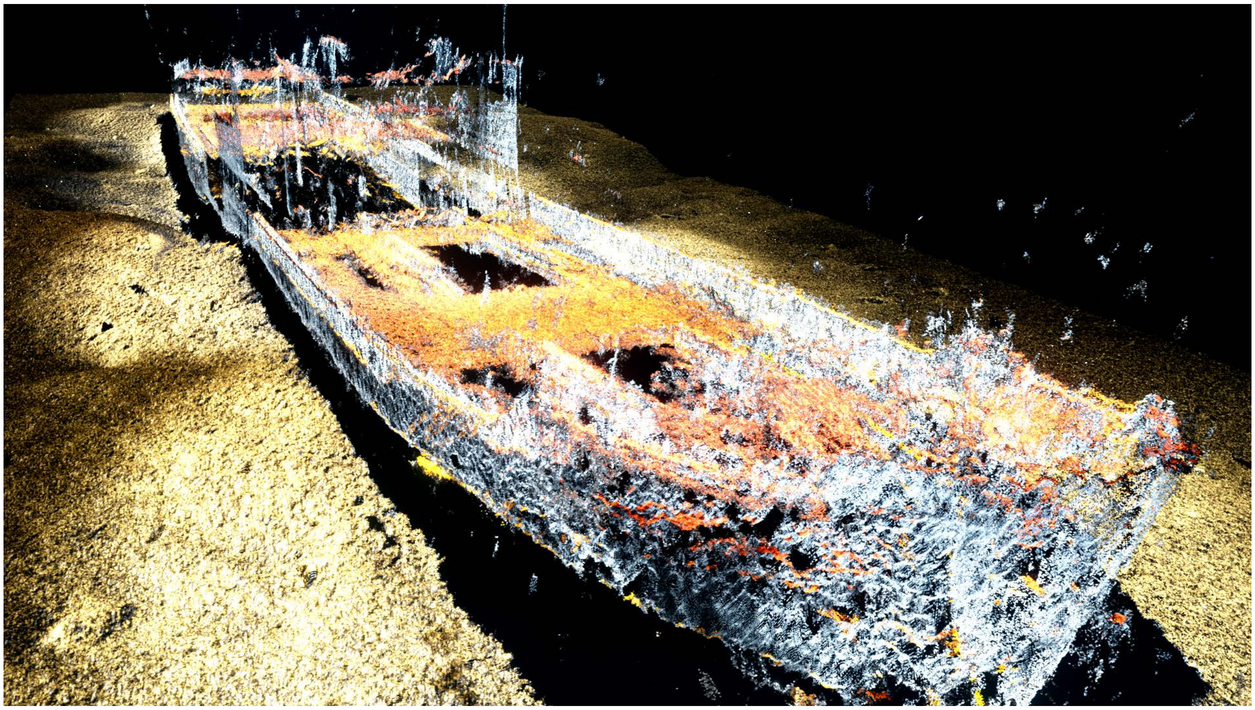

Figure 1: An example frame from one of our final 3D visualizations using the techniques discussed in this paper.

In this work we experiment with visualization of SAS data using state-of-the-art 3D rendering software. In Figure 1 we show an example frame from one of our final 3D visualizations. The goal is to provide a better, more intuitive impression of the data than traditional images. Our intention is to provide the viewer with improved understanding of the data set, which can make human data interpretation easier and thus improve the possible applications and worth of the data.

2. VISUALIZATION

With solely the SAS image data, we can produce various 2D image products. These include the following, which we show examples of in Figure 2:

Sectorscan image A single-ping, polar coordinate representation using the phased-array beam steering.[3] Can be animated along the slow-time dimension, i.e. one frame per ping position which will give an image sliding across the seafloor with the moving sensor.

Sidescan image The main beams from multiple pings stacked as a 2D image.[4,5] The image resolution is range-dependent and will vary across the image with lower resolution at longer range from the sensor.

Synthetic aperture image Similar to the sidescan image, but created by coherent combination of multiple pings for each pixel in accurate space and time signal processing.[1] Provides range-independent resolution in the entire image.

Waterfall Slightly overlapping sidescan or synthetic aperture images carefully stacked and played back falling vertically on a screen.[4,5] This is one of the most common ways of visualizing sonar data.

Mosaic Slightly overlapping sidescan or synthetic aperture images carefully tiled and stitched together to make a single image of a larger area.[4,5] Typically visualized in a north-referenced map view, as opposed to the above visualizations which are usually shown relative to the sensor orientation.

With interferometric sensors we can also estimate a high resolution bathymetric or seafloor depth map, either with sidescan processing, or in even higher resolution with synthetic aperture techniques. We show an example depth map and coherence (data quality estimate) in Figure 3. With the depth information we can perform proper transformations so that the 2D images listed above are correctly referenced to a ground-plane coordinate system. We show an example of this transform in Figure 4. This has a significant impact on data analysis, but also for making the images appear sharp. An example of this is a vertical wall structure, which would be stretched out over a larger area in the image, but is compressed into a sharp edge with the depth information available. However, the data points along that wall would be lost in such a transform if shown as a ground-referenced 2D image, so we lose some information by doing so while gaining an image that can be easier to interpret.

Furthermore, with the depth map we have the ability to perform these additional visualizations:

Depth map 2D image The depth map can also be visualized as a color-coded 2D image.

Fusion methods The backscatter and depth data can be combined using various methods into a pseudo-color 2D image.[6]

3D surface plot The depth map can be used to estimate a surface representation of the seafloor. This surface can furthermore be colored either by the depth data, or by the backscatter or fusion data.

3D scatter plot The depth map data points can be visualized as separate data points in a scatter plot. The data points may be colored either by the depth data, or by the backscatter or fusion data.

As a derived product, we can also use the high resolution depth map to estimate the seafloor slope direction, i.e. normal vectors, which in turn can be used to improve the visualization.

Taking the visualization even further, we note that it is possible to introduce artistry by sculpting and data manipulation based on human analysis and input, but in this work we will only consider data products made from actual measurements or mathematically estimated from these.

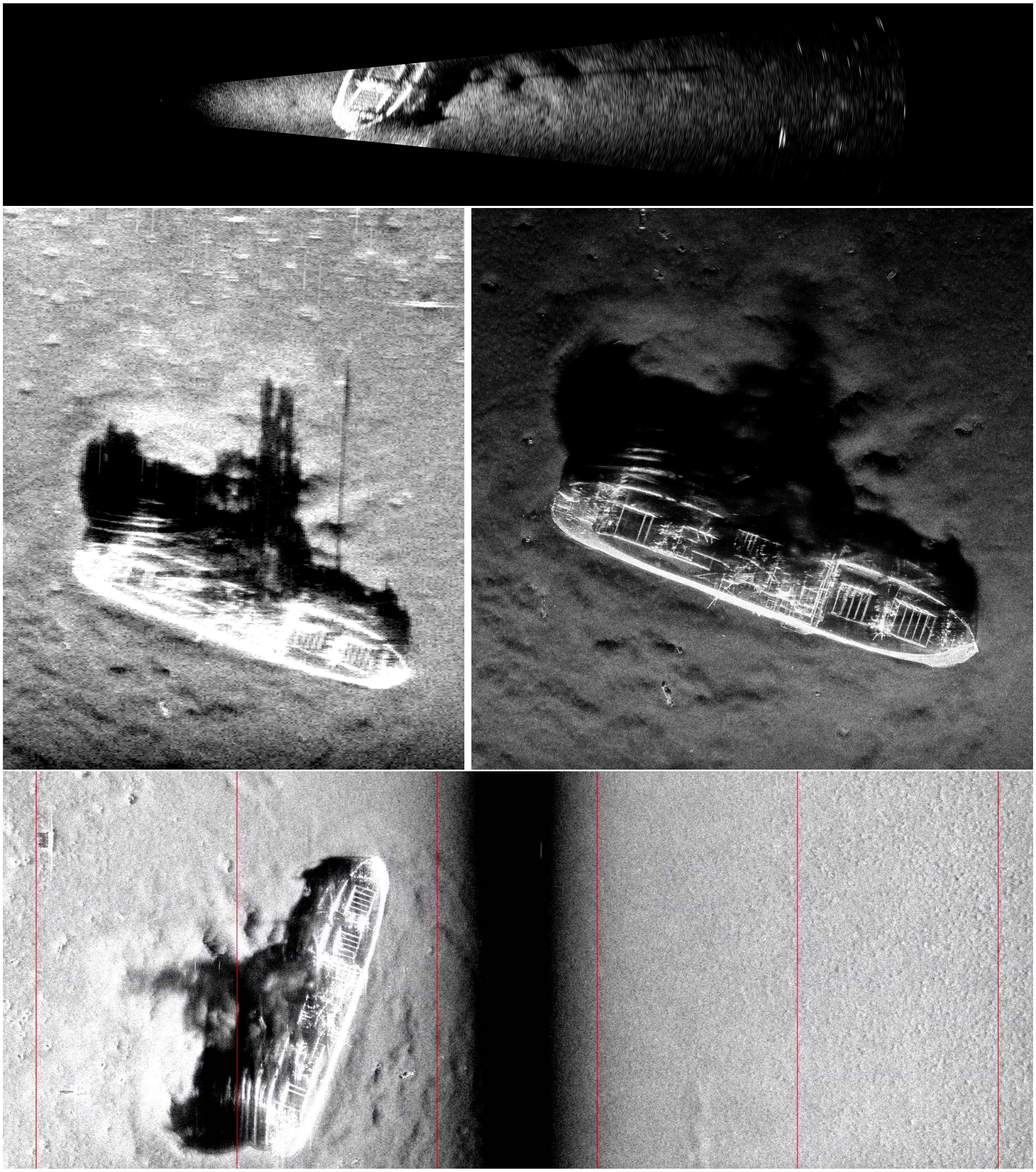

Figure 2: Image products from the SAS data. The top panel shows a sectorscan image. The middle panels show sidescan and synthetic aperture images on the left and right side respectively. The lower panel shows a synthetic aperture waterfall image with the port and starboard side, and the blind zone in the center directly underneath the sensor platform.

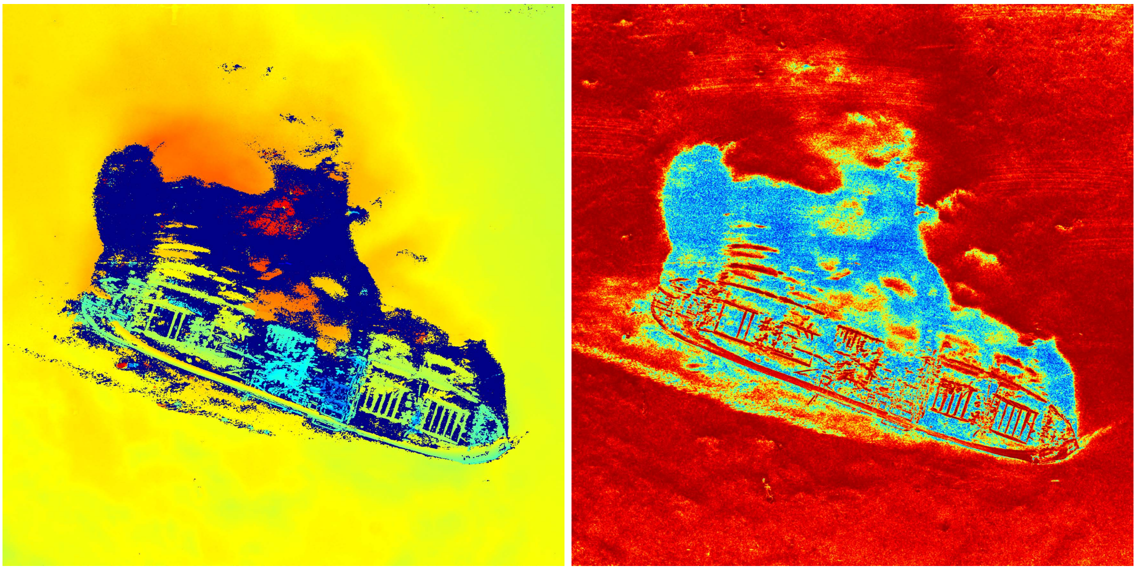

***Figure 3:** Bathymetry products from the SAS data. The left panel shows the SAS depth map and the right panel shows the corresponding coherence estimate, where red values indicate high data quality.*

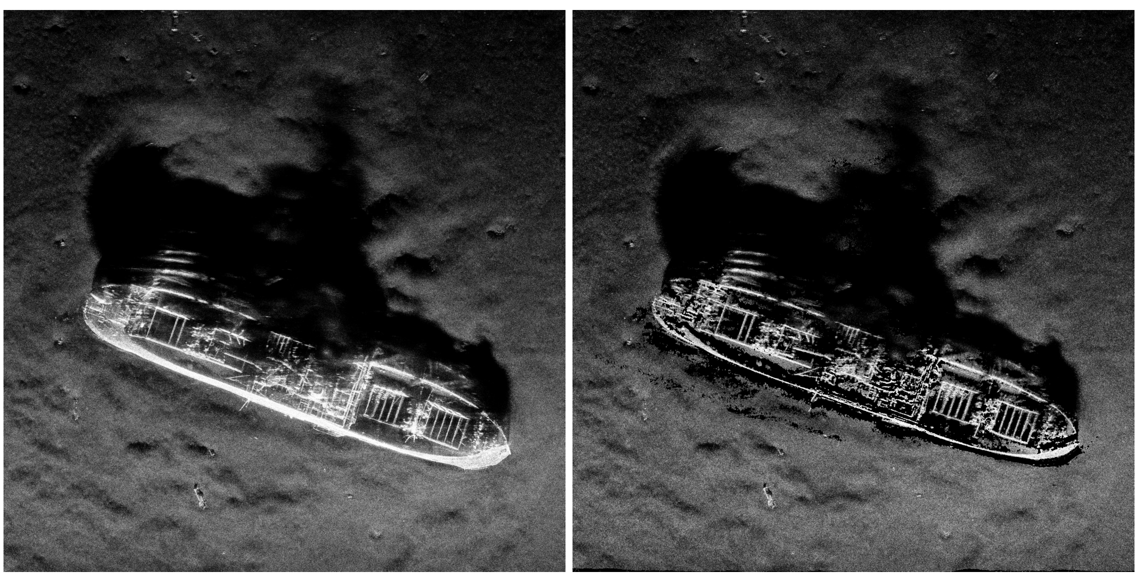

***Figure 4:** SAS 2D images without and with geometric correction in the left and right panels, respectively. The geometric correction transforms the image to a ground-plane coordinate system using the estimated depth from the interferometric processing.*

3. DATA PREPARATION

All the data presented in this paper were processed in the FOCUS synthetic aperture sonar signal processing toolbox developed by FFI,[7] and then exported to the open source 3D software Blender[8] for 3D rendering.

The SAS imaging is done using the backprojection algorithm in ground coordinates.[9]

We have used a cross-correlation based interferometric processing to estimate the interferograms and coherence maps.[10] We then used a cascaded local median-filter to unwrap any residual phase wraps in the interferograms.[2] A gradient estimator applied to the resulting bathymetric maps provides the local slope of the seafloor, which we use to orient the vertices in the visualization. Additionally, we use a wavelet-based multi-scale variance estimate derived from[11] to increase the image intensity for image pixels belonging to small scale structures such as munitions. The bathymetries, the coherences, the SAS images and the gradient maps are then all geometrically corrected for range displacements due to incorrect rendering in the imaging, resulting in a 3D point cloud with measurements rather than regularly gridded images. Wild-points detected in the bathymetric map are then used to semi-automatically clean the data.

From a single pass of a large object, like a wreck, there will be regions with missing data due to shadowing. We have therefore merged two passes, each from either side along the length of the wreck, before rendering. The merging is an automated non-coherent coordinate transform applied after the interferometric processing. A final small-scale translation in the horizontal plane is currently performed manually before the datasets are merged.

4. VISUALIZING SONAR DATA IN BLENDER

In this work we have used the open source 3D software Blender[8] for 3D rendering. Blender is intended for creating 3D animated films, visual effects and art, but is in our opinion equally useful for scientific visualization.

We initially considered creating surface representations of the sonar data. This proved difficult because of issues with creating a continuous surface in 3D and texturing it (using the data) in a way that is aesthetically pleasing and useful to the viewer. This is mainly because of the sample space being sparse in 3D, while we would prefer it to be densely sampled in 3D, or rather uniformly sampled along the surface orientation. However, this is not the case unless the scene is very simple, e.g. a close to flat seafloor - in which case the 3D visualization would not be very interesting. We specifically had issues with rapid vertical changes and interpolation in areas not densely populated with data points. We tried to create methods to form the surface by connecting vertices by 3D euclidean distance, but any noisy samples will impact the surface heavily and made it difficult to achieve a satisfactory result without significant manual processing. Heavy smoothing would be one way to mitigate this issue, but we desire to keep the details in the measurements instead of smoothing them out, and thus we don't consider heavy smoothing a useful solution.

Following this insight we investigated methods to visualize the data as point clouds, which circumvents the before mentioned issue of noise samples because we are not drawing a surface to each sample. Instead, each sample is rendered individually with free space surrounding them. In this case, a noise sample will not be as pronounced.

To achieve point cloud rendering, we first considered vertex instancing in Blender. This is a method in which an arbitrary object is inserted into each vertex position of a mesh. We show an example of this in Figure 5. This fits our need, but with a large number of vertices, the implementation in Blender was slow performance-wise and frequently crashed in our experience. Furthermore, we did not find any straightforward way to add vertex colors to the instanced objects. Again, this is a matter of the implementation because Blender does not import vertex colors on meshes without edges or surfaces. This may change in future versions as Blender is currently under very active development.

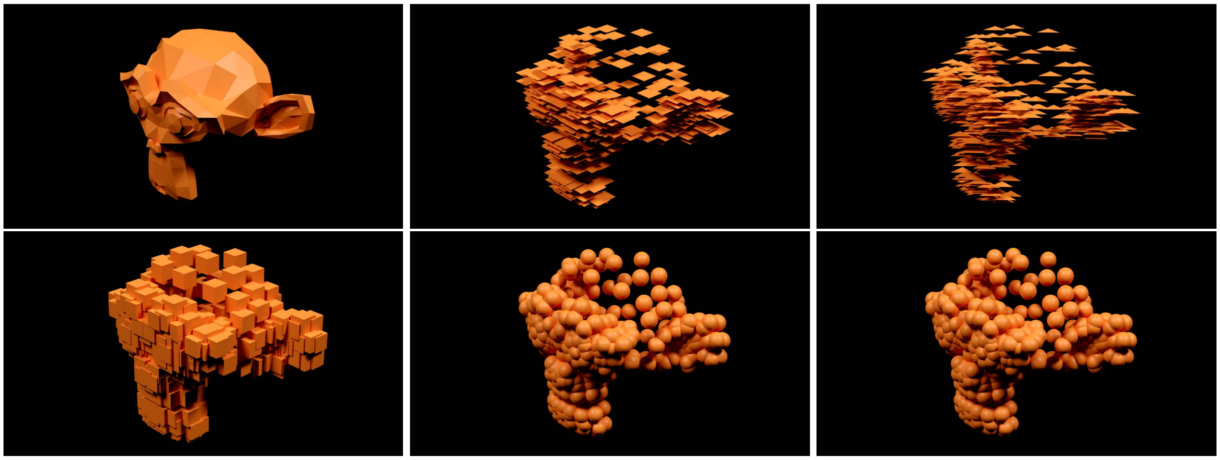

Figure 5: Example of vertex instancing with Suzanne the monkey, the default test object in Blender. Reading from the top left panel, we show Suzanne the monkey surface representation, and then vertex instancing with planes, triangles, cubes, spheres, and smooth shaded spheres.

To overcome these limitations we implemented our own version of vertex instancing in Matlab. We insert an arbitrary object in the position of each vertex, and add vertex colors to it. Furthermore, we implemented the possibility to rotate the object by a given rotation angle. We used the normal vector of the seafloor for this rotation angle, which was estimated directly from the high resolution depth map.

To move the new and quite large data set between Matlab and Blender we used the Stanford PLY format.[12] The PLY format contains an ASCII header followed by lists of vertices and faces in either ASCII or binary format. Using the binary format saves a significant amount of space used for storing the data. For Matlab this is less significant other than the time it takes to write the data to disk, but the space saving was required in order to enable Blender to read the files without crashing, even when enough computer memory is available. For large datasets consisting of more than a few tens of millions vertices we also made a method to split the dataset into multiple file parts so that these could be imported one by one into Blender, and then joined later using Blender's internal join mesh functionality. Albeit not necessary, joining the meshes simplifies the setup for further 3D rendering.

Choosing the object to render as points in the point cloud is a significant decision. We illustrate possible alternatives in Figure 5, including planes, triangles, cubes and spheres. We also considered trihedras and low poly spheres, i.e. spheres with a small mesh making it appear less smooth. Because of performance considerations with very large meshes, we ended up using triangles. It is worth mentioning that while spheres may seem like a good option because they will be equally visible from all directions when moving around in a scene, it is also not possible to incorporate data such as the seafloor normal direction, since rotating a sphere will not show any effect. If we did not have to use triangles because of performance considerations, we found that we got nice renders using cube objects.

With the data set now represented as point clouds of rotated triangle objects, we add color to them using a fusion of backscatter intensity and color from the depth map. We add sunlight for fill-lighting the scene, and omnidirectional light and spotlight sources for emphasizing features in the scene. Furthermore, we use volume rendering with an absorption effect to mimic an underwater environment. We illustrate these steps in Figure 6.

With the scene set, we add a camera. One of the main advantages of using Blender for this visualization is that we have a virtual camera that can mimic a traditional camera and create realistic effects. We can use aperture to control the depth of field, control the focus point, and choose the focal length and camera position to our liking. We have illustrated some of these in Figure 7. Furthermore, we can use camera movement

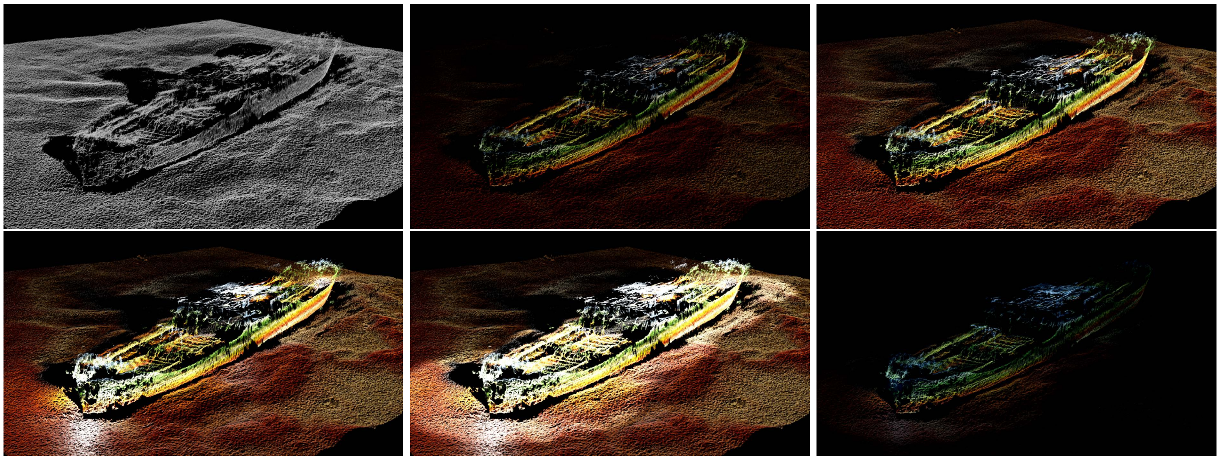

Figure 6: Example of the buildup of a 3D rendered scene. Starting from the top left panel, we show the point cloud data as triangles without any color added, then we add color, add sunlight, add omnidirectional lighting inside the wreck, add overhead spotlights emphasizing the wreck, and finally add absorption volume scattering to mimic an underwater environment.

to explore the data or to emphasize parts of the data set. We illustrate a camera movement in Figure 8, but strongly encourage the reader to look up the video presentation to get an impression of the effect of this.[13,14]

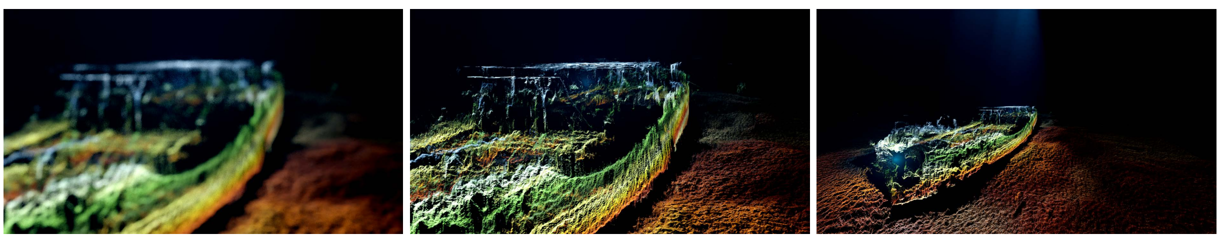

Figure 7: In the left panel we show a render with the wreck out of focus, then the same render, but in focus in the middle panel. In the right panel we show the render with the exact same camera position, but with a focal length corresponding to a wide-angle lens.

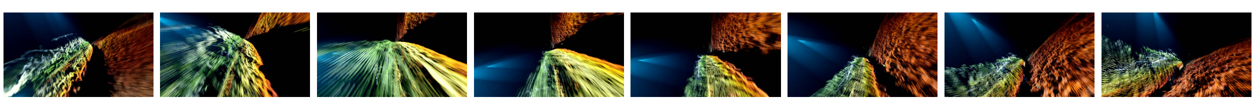

Figure 8: A storyboard-like timeline of active camera movement with position change in both x, y and z, as well as rotation around the same axes. Also notice the significant motion blur that has been added to make the movie look intuitively more realistic, but also making the camera movement less staggering and artificial.

At this point, one of the main advantages of using a 3D rendering software such as Blender becomes apparent: We can now use any techniques, models or effects from 3D rendering and animation and apply it on our visualization. We illustrate a simple example in Figure 9 by adding bubbles flowing in the scene to create some vivid movement in the static scene, but the possibilities are endless.

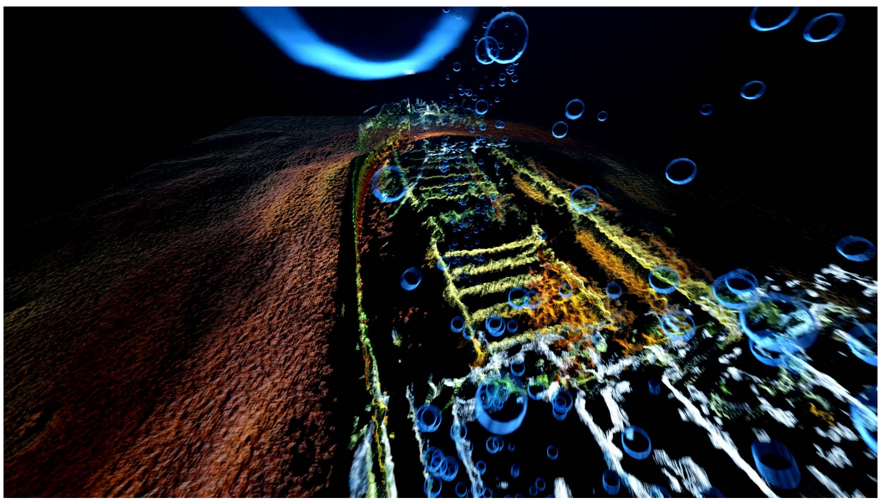

Figure 9: The addition of special effects illustrated by simple bubbles floating up through the water column through the scene. The bubbles are rendered as spheres with glass surfaces, so they exhibit refraction of the light passing through them.

5. FINAL RENDER

Putting all this together we show another example of a final render in Figure 10, in addition to the one in Figure 1. These renders were prepared for a digital and a physical travelling exhibition with the German Maritime Museum,[15] putting attention on the issue of toxic remnants of war that have been dumped in the North Sea area. The 3D visualization enables animated videos that can be watched and better understood by a broader audience. We show animations for these in the video presentation[13,14] for both the wreck in Figure 1[16] and the one in Figure 10.[17]

Apparent in Figure 1 is the absence of data points on surfaces that is oriented across the beam of the ship - in particular on the wheelhouse, which is practically see-through in the render. This is not due to hull damage on the wreck itself, but the imaging geometry we have chosen running along the length of the ship. If we had also included perpendicular passes or in other directions, we could fill in these data points. The cost of doing so would be that data coregistration becomes more difficult.

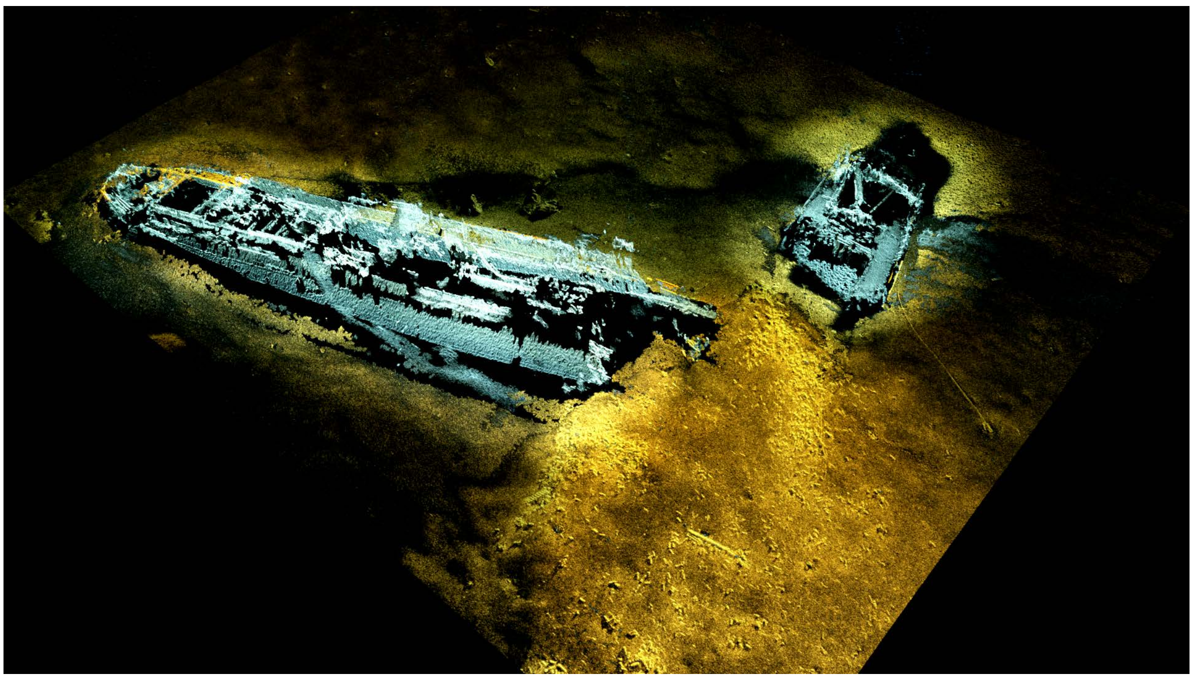

Figure 10: Example of a final render of a shipwreck belonging to a chemical munitions dump site in the Skagerrak Sea. Munitions can be seen scattered from the broken wreck.

6. CONCLUSION

We have constructed a work flow for visualizing interferometric SAS data as point clouds using 3D rendering. We suggest to incorporate both the backscatter, depth and quality data into the coloring, and to incorporate extra information by rotating each instanced object by the estimated slope of the seafloor. Furthermore, we have discussed 3D rendering visualization techniques such as camera control and special effects. We have included still frames of rendered results here, while the resulting 3D animations can be seen in the conference video contribution.[13,14,16,17]

ACKNOWLEDGMENTS

We would like to thank the Norwegian Coastal Administration and the crew and scientists on the Norwegian Defence Research Establishment's (FFI's) research vessel H. U. Sverdrup II for their contribution in collecting the data.

REFERENCES

[1] R. E. Hansen, "Introduction to synthetic aperture radar," in *Sonar Systems*, N. Z. Kolev, Ed. IntechOpen, 2011, ch. 1. [Online]. Available: https://doi.org/10.5772/23122

[2] T. O. Sæbø, "Seafloor Depth Estimation by means of Interferometric Synthetic Aperture Sonar," Ph.D. dissertation, University of Tromsø, 2010.

[3] J. L. Sutton, "Underwater Acoustic Imaging," *Proc. IEEE*, vol. 67, no. 4, pp. 554–566, April 1979.

[4] P. Blondel, *The Handbook of Sidescan Sonar*, ser. Geophysical Sciences. Springer Praxis Books, 2009.

[5] J. P. Fish and H. A. Carr, *Sound reflections: Advanced Applications of Side Scan Sonar*. LowerCape Publishing, 2001.

[6] Ø. Ødegård, R. E. Hansen, H. Singh, and T. J. Maarleveld, "Archaeological use of Synthetic Aperture Sonar on Deepwater Wreck Sites in Skagerrak," *Journal of Archaeological Science*, vol. 89, p. 113, January 2018.

[7] R. E. Hansen, T. O. Sæbø, H. J. Callow, P. E. Hagen, and E. Hammerstad, "Synthetic Aperture Sonar Processing for the HUGIN AUV," in *Proceedings of Oceans '05 Europe*, vol. 2, Brest, France, June 2005, p. 10901094.

[8] Blender Foundation, "Blender 2.92.0," https://blender.org, accessed: 2021-08-27.

[9] R. E. Hansen, H. J. Callow, T. O. Sæbø, and S. A. V. Synnes, "Challenges in Seafloor Imaging and Mapping with Synthetic Aperture Sonar," *IEEE Trans. Geosci. Remote Sensing*, vol. 49, no. 10, p. 36773687, October 2011.

[10] T. O. Sæbø, S. A. V. Synnes, and R. E. Hansen, "Wideband interferometry in synthetic aperture sonar," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 8, pp. 4450–4459, 2013.

[11] M. Geilhufe and Ø. Midtgaard, "Quantifying the complexity in sonar images for MCM performance estimation," in *Proc.* 2^{nd} *UACE*, Rhodes, Greece, June 2014, pp. 1041–1048.

[12] P. Bourke, "PLY - Polygon File Format," https://web.archive.org/web/20210407105514/http://paulbourke.net/dataformats/ply/, accessed: 2021-04-07.

[13] O. J. Lorentzen, R. E. Hansen, T. O. Sæbø, S. A. V. Synnes, and M. Geilhufe, “3D Rendering of Shipwrecks with Synthetic Aperture Sonar,” https://web.archive.org/web/20210831161344/https://www.uaconferences.org/component/contentbuilder/details/26/123/uace2021-3d-rendering-of-shipwrecks-with-synthetic-aperture-sonar?Itemid=504&start=0, accessed: 2021-08-31.

[14] O. J. Lorentzen, R. E. Hansen, T. O. Sæbø, S. A. V. Synnes, and M. Geilhufe, “Youtube: 3D Rendering of Shipwrecks with Synthetic Aperture Sonar,” https://youtu.be/J7rNeKaivkg, accessed: 2021-08-31.

[15] German Maritime Museum, “Toxic legacies of war - North Sea wrecks,” https://web.archive.org/web/20210827110757/https://www.dsm.museum/en/exhibition/exhibitions/toxic-legacies-of-war-north-sea-wrecks, accessed: 2021-08-27.

[16] O. J. Lorentzen, R. E. Hansen, T. O. Sæbø, S. A. V. Synnes, and M. Geilhufe, “Youtube: 3D Rendering of Shipwrecks with Synthetic Aperture Sonar - Wreck 18,” https://youtu.be/J7rNeKaivkg?t=794, accessed: 2021-08-31.

[17] O. J. Lorentzen, R. E. Hansen, T. O. Sæbø, S. A. V. Synnes, and M. Geilhufe, “Youtube: 3D Rendering of Shipwrecks with Synthetic Aperture Sonar - Wreck 5,” https://youtu.be/J7rNeKaivkg?t=708, accessed: 2021-08-31.