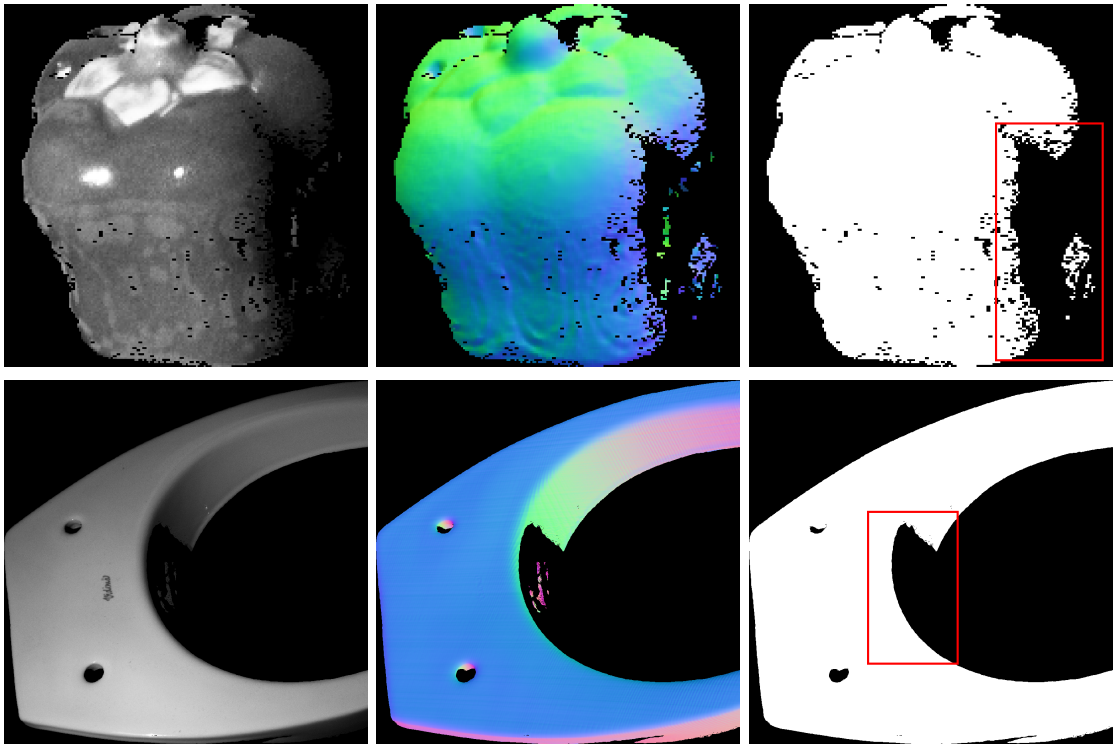
3D Scan filtering

1 Input data



(a) intensity map (b) normal map (c) ground truth

The input is a scan from the Photoneo 3D scanner, which is an point-cloud organized into a 2D matrix. It contains depth, intensity and normal of each point. We assume that all of the training examples consists of objects from a single, homogeneous material - glossy ceramics. The distribution and characteristics of artefacts (mainly caused by reflections) should be therefore similar.

As Figure 3. exemplifies, even though the points may look correct on the single view scan, their location (depth) is in-fact way off the object's surface (notice the illustrative red frame in ground truth). Please note, that from a single scan, we can possibly generate many training/test examples, by using a sliding window. i.e. cutting the scan into patches [2].

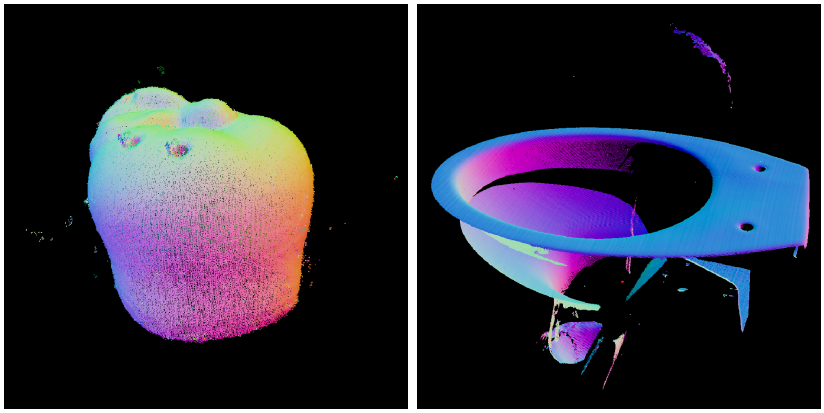


Figure 3: 3D rendering of the objects, fake geometry is more notable.

2 Problem formulation

Given an input scan matrix of size $n \times n$, where n is a modifiable hyper-parameter (can prepare windows of any size), classify each point based on whether it is a true geometry, or an error/artefact. The ground truth is created either by hand, or using an existing (non-ML) filtration. This algorithm needs the full, reconstructed object, as it uses the redundancy between scans (overlays). That is one of the drawbacks of the existing solution and our ML model should overcome this by being able to filter a single-view scan. The final labeling can be visualized as a simple segmentation mask, where true points are white.

3 Proposed methods/ideas

At first, I want to spend some time with feature extraction, as the absolute values of intensities/normals might not be the best. For example, for each point c, we may consider its neighboring points $o^i; 1 \leq i \leq k \times k$; where k is some fixed neighborhood size. Then, we may calculate $avg_i \, |c_{depth} - o^i_{depth}|$. Features like these specify, how different the point is from its neighborhood and the model might find patterns here.

Using a neural network with convolution layers is the sort of generic idea for a problem like this. However, even different architectures and simple MLP layers proved to be effective [1]. Since it is a classification, I definitely plan to experiment with some SVM aswell.

U-shaped neural networks seem to work-well for similar problems [2]. I was also thinking about training two parallel networks. One would consider bigger, more global neighborhood (possibly the whole scan) and the second just small, local neighborhood. After few layers, feature maps would con-cat and the branches would connect back to a single network before outputting a final classification/labeling. I am considering the case, where we would run the network individually for each point of the scan, but that might be quite slow. Thinking...

Overall, I will be thankful for feedback and ideas. This problem is a part of my master's thesis.

References

[1] R. Charles, Hao Su, Mo Kaichun, and Leonidas Guibas. PointNet: deep learning on point sets for 3D classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, 07. 2017.

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: convolutional networks for biomedical image segmentation. volume 9351, pages 234–241, 10. 2015.