376.054 Machine Vision and Cognitive Robotics
Exercise 5: Plane fitting with RANSAC

Severin Jäger, 01613004

December 22, 2020

Contents

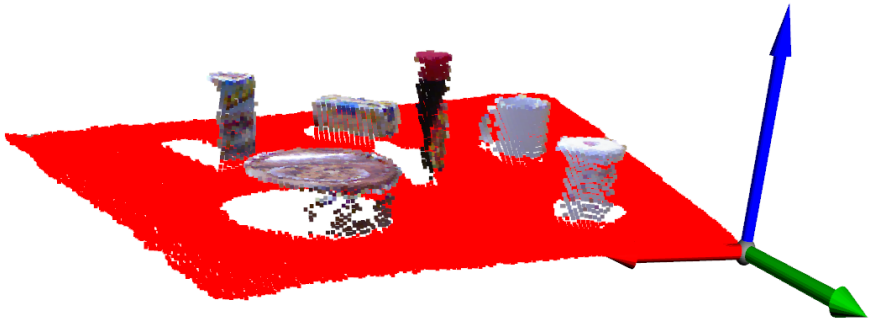


Figure 1: Ground plane detection result with a confidence of 0.9, an inlier threshold of 2 cm, and a minimal sample distance of 80 cm

1 Ground plane detection results

In the first step, the basic RANSAC algorithm was used to detect the ground plane in an image as shown in Figure 1. The plane was clearly detected, this is a reasonable start for segmenting the objects on top of the plane. However, this is a rather simple point cloud consisting basically of one plane and a few small objects atop.

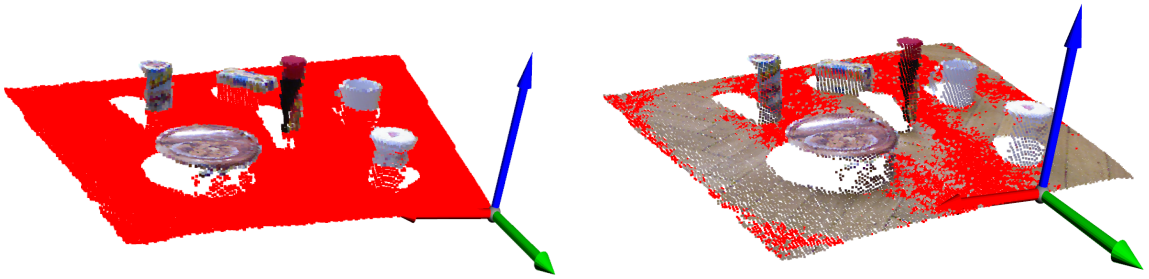
2 Influence of RANSAC parameters

The RANSAC implementation used above has three major parameters: The inlier threshold which determines the maximum distance between the plane and points to be considered inliers, the minimal sample distance, which determines if a random sample is discarded due to the vicinity of the points and the confidence, which influences the termination of the RANSAC algorithm.

Figure 2 (a) shows exemplarily a too high value of the inlier threshold. As a result, major parts of the objects are detected as part of the ground plane. This reduces the probability of correct segmentation an classification of the objects. However, a too small value can lead to significant problems as well. As shown in Figure 2 (b), this yields planes not containing all relevant points due to sensor noise.

The minimal sample distance has less influence on the quality of the detected plane. In rare cases, it is possible that very close points lead to wrong planes, however the algorithm will find better results in later iterations.

Tuning the confidence does not only affect the quality of the fitted plane, but also the runtime of the algorithm. On one hand, a sufficient level is confidence is required for proper detection (s. Figure 3), on the other hand high values increase the number of iterations and thus the runtime. Therefore, this parameter involves a trade-off between accuracy and complexity.



(a) (b)

Figure 2: Ground plane detection with the parameters used for Figure 1, but an inlier threshold of 5 cm (a) and 2 mm (b).

Figure 3: Ground plane detection with the parameters used for Figure 1, but a confidence of 0.12

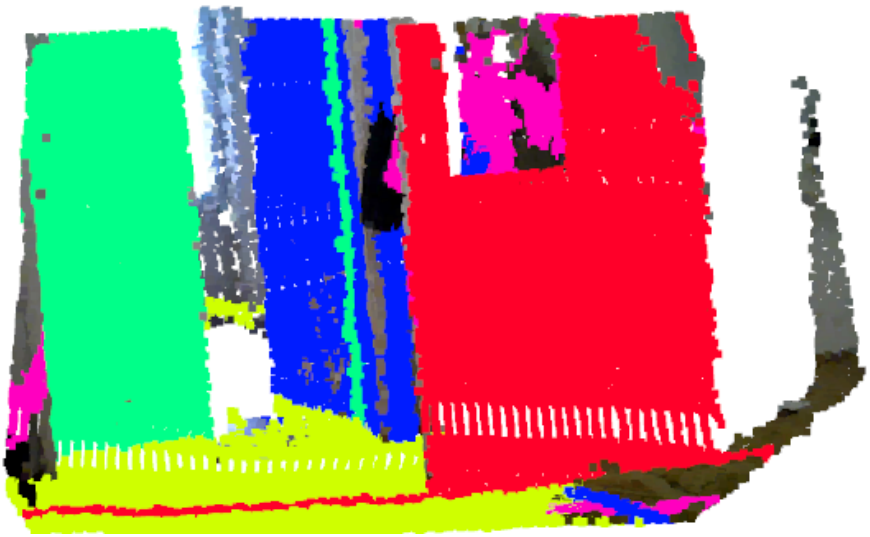


Figure 4: Detected planes in the door point cloud with a confidence of 0.9, an inlier threshold of 5 cm, and a minimal sample distance of 30 cm

3 Multi-plane detection results

This section deals with detection of multiple planes in point clouds. As before, the basic RANSAC error function was used. Figure 4 shows the 5 detected planes in the door point cloud. The most dominant planes (door in the front in red, floor in yellow, left open door in blue, closed door in green) are detected nicely, however there are some interesting artefacts. Firstly, points belonging to the floor are mapped to the door in the front as this is the largest (and therefore initially detected) plane. The blue plane shows a similar behaviour. Furthermore, the pink plane does hardly belong to any physical plane, but is the best fit for the remaining data points (after removing the dominant planes). This can be avoided by increasing the number of points a plane needs to be considered.

In Figure 5, the planes detected in the desk point cloud are shown. Here the desk plane is detected correctly and the three screens are detected as two planes. However, the algorithm detected a plane connecting the different structures in the background which is likely to be an artefact caused by the viewpoint and the FOV. However, this plane again contains only a few data points.

The results of the plane detection in the most complicated scene are shown in Figure 6. The most important planes are still detected properly, however there are planes just fitting the points nicely but not corresponding to a physical plane as well as data points matched to a plane without belonging to it (e.g. the items in the dishwasher). These two effects are hard to mitigate, one approach might be discarding planes consisting of points being far apart, however this might lead to problems if other objects cover parts of the plane. In general, different image structures require different parameters, this makes is hard to implement a well-scaling detection.

All images shown so far used a straightforward uniform down-sampling approach to reduce the computational complexity. This leads to unpleasant artefacts such as the highly regular holes in the lower part of Figure 4. However, there are fancier techniques such as voxel-based down-sampling are available. This approach leads to smoother point clouds, nonetheless the detection results are not significantly improved.

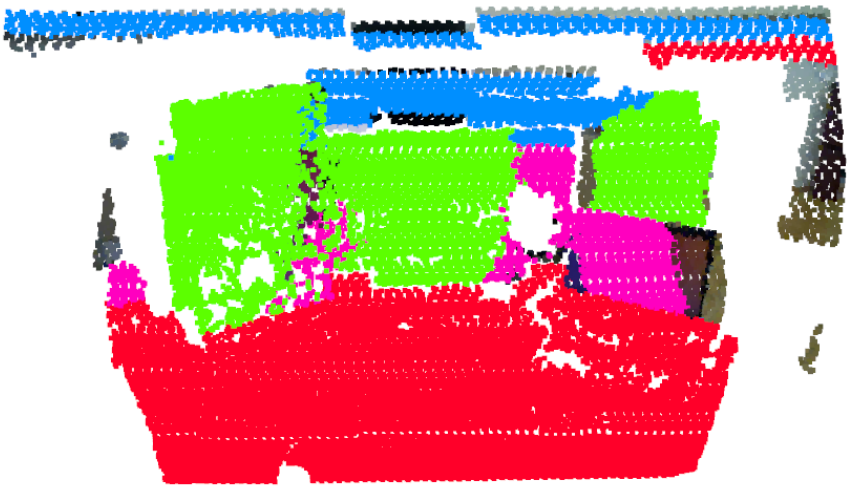


Figure 5: Detected planes in the desk point cloud with a confidence of 0.9, an inlier threshold of 5 cm, and a minimal sample distance of 30 cm

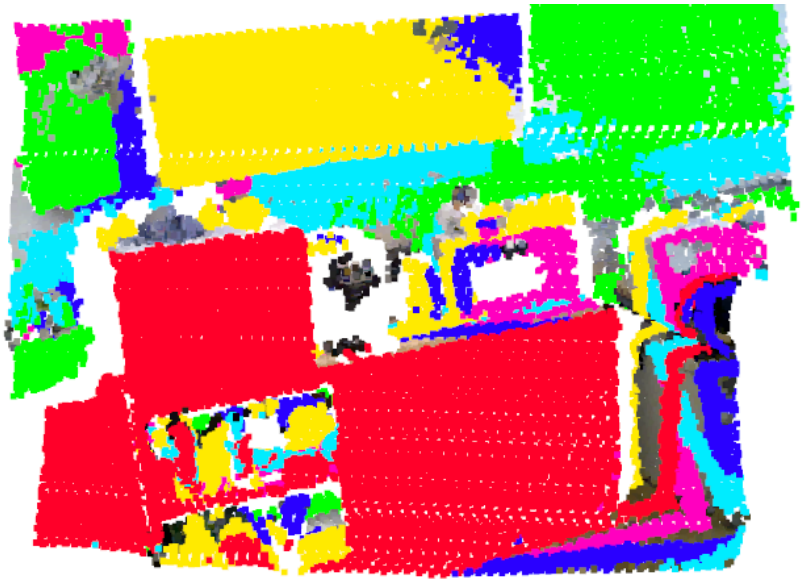


Figure 6: Detected planes in the kitchen point cloud with a confidence of 0.9, an inlier threshold of 5 cm, and a minimal sample distance of 30 cm

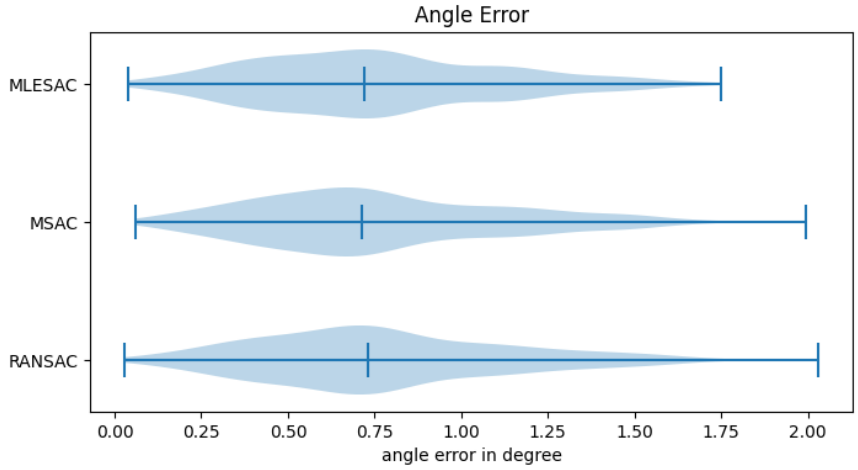


Figure 7

One further remark regarding RANSAC has to be made: The detection results might strongly differ between consecutive experiments. This comes from the randomness of the algorithm. It might come op with one plane in the first try and another one (with a similar number of points) in the next attempt. This particularly happens if there are no obvious planes in the point cloud.

4 MSAC and MLESAC

RANSAC follows a very simple approach and gives the error function as number of inliers. This leaves room for improvement by assigning each point a weight depending on its proximity to the estimated plane. MSAC and MLESAC follow this approach.

To compare the performance of these algorithms, the following experiment was conducted: A plane was generated and noise with $\sigma = 25\,\text{cm}$ was added. Then the best plane was fitted with all three error functions 1000 times respectively with a confidence of 0.9, an inlier threshold of 2 cm and a minimum sample distance of 80 cm The distributions of the resulting angular and z-direction errors are shown in Figures 7 and 8. The first observation is that the differences between the approaches is rather small. Despite the significantly increased complexity of the algorithm, the performance of MLESAC is just slightly better than what the simpler approaches achieve. Nonetheless, the average angular and z-direction error of MLESAC is the smallest and MSAC performs better than RANSAC in both metrics. So from the accuracy point of view, there is no point to use RANSAC in favour of the (similarly easy to calculate) MSAC. In case the estimation accuracy is of utmost importance, MLESAC can be considered.

This observation is also backed by the fact, that the convergence speed (expressed in number of iterations) hardly differs (s. Figure 9). Again, MLESAC and MSAC perform slightly better, however the increased computational complexity (mainly of MLESAC) is likely to compensate this.

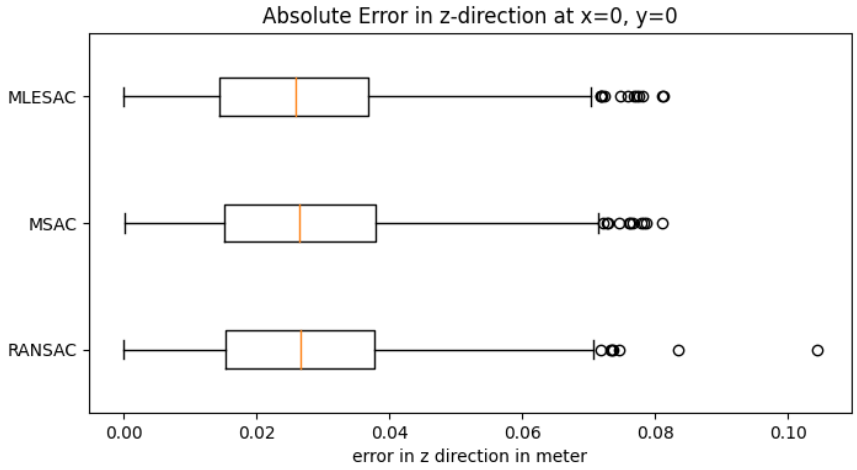


Figure 8

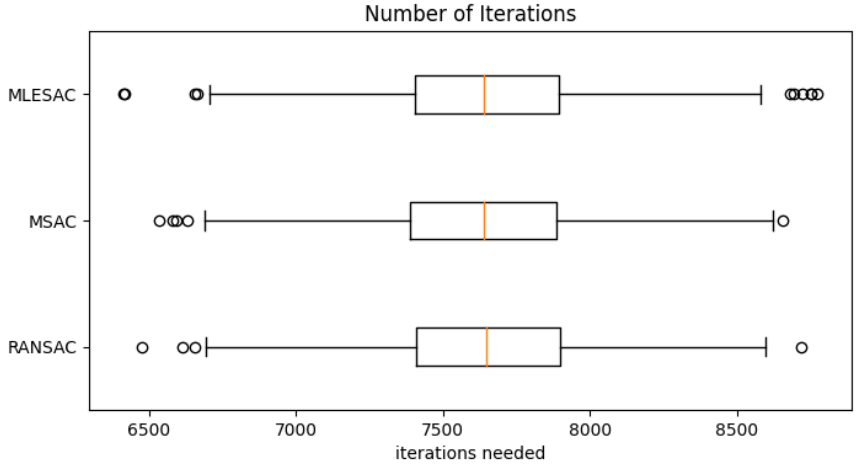


Figure 9