Process of Point Clouds Merging for Mapping of a Robot's Working Environment

Petr Olivka$^{(\boxtimes)}$, Michal Krumnikl, Pavel Moravec, and David Seidl

Department of Computer Science, FEECS, VŠB – Technical University of Ostrava,
17. listopadu 15, Ostrava, Czech Republic
{petr.olivka,michal.krumnikl,pavel.moravec,david.seidl}@vsb.cz
http://www.cs.vsb.cz

Abstract. The lidar is nowadays increasingly used in many robotic applications. Nevertheless the 3D lidars are still very expensive and their use on small robots is not economical. This article briefly introduces a construction of cheap 3D lidar for indoor usage based on the 2D laser range finder. Subsequently, this article introduces process of merging acquired point clouds. The every pair of neighboring point clouds is oriented in space to fit together in the best possible way. The result of this process is a 3D map of the robot working environment. This map can be segmented and further used for a navigation. In this way, it is also possible to map inaccessible and dangerous areas.

Keywords: Lidar · Laser range finder · Point clouds · Mapping · Robot

1   Introduction

Nowadays there are many types of laser devices for a distance measurement. Probably the best known application of this type of devices is the use of lidar in the autonomous car driving systems. For example, Google cars use lidar Velodyne HDL-64E which can be seen in Fig. 1.

It is not possible to use this type of lidar on small robots in industrial applications. There are many reasons for not using Velodyne lidar. At first, it is very expensive. The Velodyne HDL-64E price is around $8000. For small robotic applications such price is absolutely unacceptable. The next reason is the narrow measuring range. Despite this lidar measures 360° around, the vertical field of view is only 28° [12]. The visualization of the raw data from HDL-64E is in Fig. 2. The small viewing angle is suitable in traffic environment, but it is insufficient for indoor application like recognition of nearby objects and obstacles. But what is for internal use completely inappropriate is the blind space above the lidar. For the indoor robot navigation it is very important to detect space above the robot.

In contrast to this high-end devices there are several cheap devices available on the market suitable for home or office use. The first example is Kinect as depicted
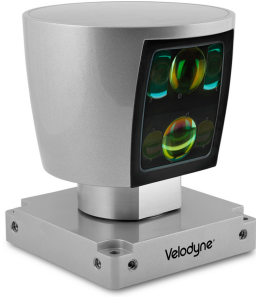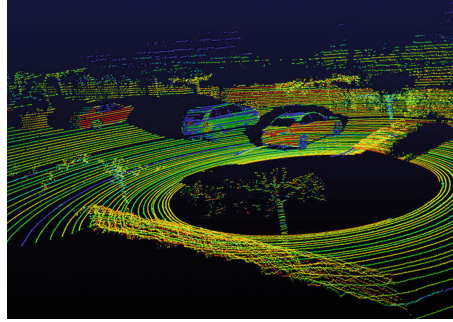
Fig. 1. Velodyne lidar HDL-64E [12]   **Fig. 2.** Raw data from HDL-64E [12]

in Fig. 3 and a similar device Xtion PRO can be seen in Fig. 4. The disadvantage of these devices is that they are not designed for industrial applications and they have very limited measurement field of view in vertical and horizontal axes. These devices are also not prepared for measurement in 360° range.

Because 3D lidars are nowadays still very expensive and cheap customer devices are not suitable for industrial usage, it is necessary to focus to a different device design for a reliable measurement of distances in 3D space. There is a large variety of 2D Laser Range Finders (LRF) available on the market. These devices are produced for measurement of short or long distances with different measurement precision, they are equipped with different interfaces and they are designed for indoor or outdoor use. These 2D devices can be easily used for the measurement in 3D. It is only necessary to add the third dimension. The probably first design of 3D LRF device was introduced in 2009 and it can be seen in Fig. 5. In this figure there is 2D LRF Hokuyo URG-04LX mounted on a servo in order to achieve the ability to measure in 3D.

This design has two disadvantages. The servo has a problem with precision of positioning. It is impossible to achieve the precise angular position and the repeatability of the positioning is also problematic. The second problem of this design lies in the mechanical features of the servo. The design of the servo does not handle axial and radial forces correctly. The use of this design on the robot will let to a quickly damage of the servo. The inertial forces of mounted 2D LRF will very quickly damage gears and their mounting.

Fig. 3. Kinect [http://www.microsoft.com]   **Fig. 4.** Xtion PRO LIVE [http://www.asus.com]

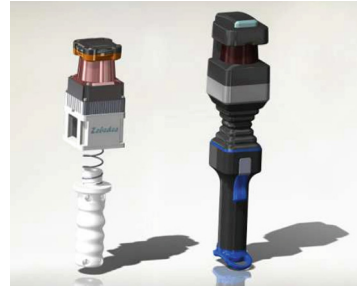**Fig. 5.** First known design of 3D LRF with servo [10]

Fig. 6. Zebedee design of 3D LRF with 2D LRF mounted on spring [1]

Alternative design of 3D LRF was introduced by SCIRO [1]. In this design the 2D LRF is mounted on a spring and the third dimension is added by a swinging. The overall design is shown in Fig. 6. Such design has also some disadvantages. It is necessary to manually swing with the mounted LRF and maintain very precise synchronization between the gyroscope mounted under the LRF and the LRF scanning.

2 3D LRF Desing

Based on the introduced constructions and their disadvantages we have developed a new design of the 3D LRF. The diagram of this design is in Fig. 7. The 2D LRF Hokuyo URG-04LX is mounted on a step motor by a carrier. This construction guarantees the sufficient strength of the design. All inertial forces are captured in ball bearings of the step motor and the step motor also guarantees the precise positioning of the mounted LRF.

In the diagram we can see that the rotation axis of the step motor is identical with the internal axis x_i of the LRF. It simplifies the future computation. The current position of the step motor is given by angle α_r and the position and
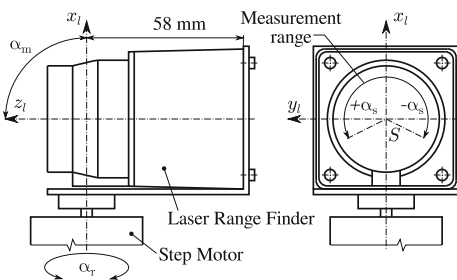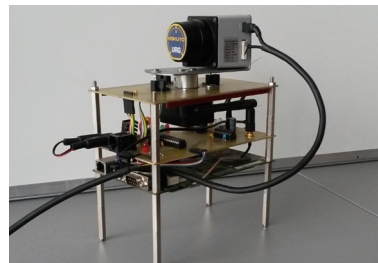

Fig. 7. 3D LFR schema

Fig. 8. 3D LRF overall design

measurement range of laser beam is defined by angle α_s. The mounting position of LRF on the step motor is marked in the diagram as the angle α_m.

Now the principle of 3D LRF measurement will be described. The 2D LRF measures distances of obstacles by a laser beam. The 2D measurement is performed in the plane $x_i y_i$. The current position of the laser beam is marked in the diagram as an angle α_s. The resolution of used 2D LRF is $360°/1024$. The angular range of the measurement is from $-120°$ to $120°$ and in this range 628 points are measured.

The carrier with 2D LRF is mounted on the step motor and is rotated around axis x_i. The rotation of the whole LRF is performed in the range of a half turn ($180°$). This rotating movement guarantees the whole coverage of the environment. The current position of the 2D LRF is marked as α_r. The resolution of the step motor is $360°/800$.

The overall design of the 3D LRF is depicted in Fig. 8. The figure shows a complete device with the control computer and the microstepping unit [8]. For the following experiment the 3D LRF was mounted on the small robotic undercarriage Roomba.

The 2D LRF measures distances in all positions and measured distances will be marked as $l_{r,s}$. This measured distances can be transformed to a position vector $\boldsymbol{l}_{r,s} = [l_{r,s}, 0, 0, 1]^T$. All measured distances can be then transformed to point cloud, where every point will be represented by its position vector $\boldsymbol{p}_{r,s}$. The computation is performed by following formula:

$$\boldsymbol{p}_{r,s} = R_z(\alpha_r) \cdot R_y(\alpha_m) \cdot R_z(\alpha_s) \cdot \boldsymbol{l}_{r,s}, \tag{1}$$

where R_y and R_z are transformation matrices for rotations.

During the single measurement the designed 3D LRF captures point clouds composed of 628×401 points. It is in total 252828 points.

Practically every manufactured LRF Hokuyo URG-04LX unit has some manufacturing inaccuracies. For the precise measurement it is necessary to calibrate every LRF. The process of calibration is described in details in [9]. Acquired results are very inaccurate without this calibration.

The set of point clouds were acquired for the following experiment. Inside the building 26 meters long corridor was selected with variety of surfaces and irregular shapes. This corridor were measured by 3D LRF four times in both directions and measurements were performed approximately at a distance of one meter. Thus 25 point clouds were acquired in every measurement. Overall 100 point clouds were acquired. These points clouds will be marked as a set of point clouds M_1 to M_{100}. Point clouds in range from M_1 to M_{25} are from the first measurement, point clouds in range from M_{26} to M_{50} are from the second measurement, etc. An example of single point cloud, concretely M_1, is visible in Fig. 9.

The following task is to merge all point clouds from every measurement into one whole.
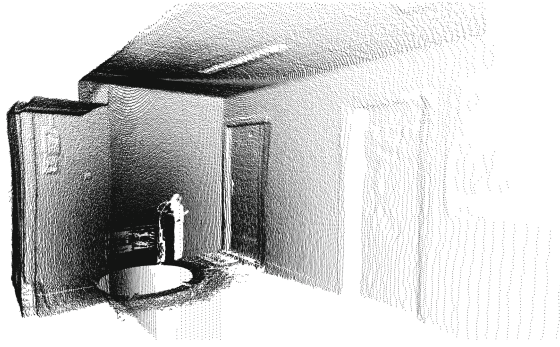
Fig. 9. Acquired point cloud M_1

3   Point Clouds Preprocessing for Merging

Nowadays there are many preprocessing methods, algorithms, filters and analysis known. But for point clouds merging only a few of them are suitable. The main focus should be concentrated mainly on these three areas:

– features,
– keypoints,
– registration.

3.1   Features

Features is general term for many characteristics of point clouds. It is possible to compute density of points, normal vector of points, collect neighbours, compute curvature of surface etc. From this list the most suitable features for preprocessing before merging are the computation of normal vectors and curvature.

In this experiment the Point Cloud Library [11] (PCL) is used. The normal vectors are in this library computed by eigenvalues and eigenvectors. The algorithm is described in more details in [2]. The knowledge of this algorithm is very important for the computation of curvature and segmentation. Without this knowledge it is not possible to understand used terms and parameters in PCL implementation.

3.2   Keypoints

The aim of keypoints is to select a subset of points from the point cloud which will represent the whole point cloud in the processing steps. The subset selection should represent the point cloud, but the reduced amount of points will increase speed of the following processing. This is similar procedure as searching for correspondence in stereometry. The keypoints detection can be performed by many algorithms. The description and the comparison and recommendation of the best algorithms after testing were introduced by Alexandre [3] and Filipe [4].

3.3   Registration

The previous algorithms are designed for usage with single point cloud and they detect important attributes in point cloud. However, the registration is directly designed for two point clouds merging. The principles of registration process based on planar surfaces are described in [5] and advanced registration for outdoor environment was introduced in [6]. The implementation of registration is based on features and keypoints.

3.4   Preprocessing Recommendation

Even there are many algorithms for point clouds preprocessing, all of them mentioned above are designed and tested for point clouds representing the small area in front of a robotic device. The point cloud acquired by the 3D LRF introduced above captures the whole environment around robotic device, not only in front. Thus problem of merging the point clouds is more complex. The merging process should be based on main principles introduced in this section, but must be designed to accept different type of point clouds.

4   Point Clouds Segmentation

The point cloud acquired by the presented 3D LRF covers big part of the environment around the device. In this situation the point cloud contains much more information than from devices which acquire only a small range of the environment. In this case it is possible to expand small details - keypoints - to larger object. Every point cloud can be divided to separated segments. The number of segments is significantly smaller than number of points in cloud. Moreover all segments can be represented by some typical features, e.g. center of gravity, normal vector, size, etc. So the usage of segmentation for point cloud preprocessing is promising and will be consequently verified.

4.1   Segmentation Algorithms

The PCL library offers a few segmentation algorithms. Most of them are described by Rusu [2].

The algorithm *Fitting simplified geometric model* is based on searching of certain structures or patterns of points. The *Basic clustering techniques* is based on space decomposition and clustering by proximity. The *Finding edges* algorithm is complementary algorithm of the next one. It is segmentation by *Region growing*. This algorithm is well known from image processing. But the implementation for point clouds must be based on different attributes. In the PCL it is based on normal vectors and curvature.

A few additional segmentation algorithms are implemented in the PCL library. The testing of all algorithms showed that the best method for segmentation is the *Region growing*. The comparison is not presented here.

Fig. 10. Point cloud M_1 after segmentation (Color figure online)

4.2   Point Clouds Preprocessing by Segmentation

The example of the segmentation of the point cloud from Fig. 9 is visible in Fig. 10. The figure contains only large segments. The threshold for the large segment was set to 500 points and every segment is distinguished by a color.

4.3   Verification of Region Growing Algorithm

Figure 10 shows that the segmentation method *Region growing* is working well for the point cloud M_1. To be sure that this segmentation can be used for the whole data set it is necessary to apply segmentation to every point cloud from data set and create overall statistic of all segmentation results.

All segments of point clouds will be separated into six groups: left, right, front, rear, top and bottom. This separation can be based on normal vector of segments. This vector can be computed in the centroid of all segments, where the centroid can be computed by the following formula:

$$c_s = \frac{1}{N_s} \cdot \sum_{i=1}^{N_s} p_{s,i}, \tag{2}$$

where c_s is the centroid of the segment s and N_s is a number of points of that segment. Vectors $p_{s,i}$ are position vectors of all points of segment s computed by (1).

The computation of normal vector of segments in PCL library automatically orients the vector direction to half-space with the origin of coordinate system. So the largest coordinate of normal vector can divide segments to their proper group. The largest positive coordinate x of normal vector specifies rear segment, the largest positive y specifies right segment, the positive z specifies bottom segment, etc.

The segmentation of all point clouds from the data set was performed and for all large segments their proper group was determined. The overall statistic is

visible in Table 1. In this table the column Minimum indicates that in all point
clouds at least 1 large top, left and right segment were recognized. The median
for these segments is 3. One bottom segment was also recognized in all point
clouds. Only front and rear segments are not recognized in all point clouds.
This is an expected result, because the measurement was performed in the long
narrow corridor.

Table 1. Statistic of the whole data set segmentation

Segments	Number of segments			
	Average	Median	Minimum	Maximum
Left	3.1	3	1	7
Right	3.3	3	1	8
Top	3.2	3	1	7
Bottom	1.1	1	1	3
Front	1.2	1	0	4
Rear	1.2	1	0	4

This statistic proved that the segmentation method *Region growing* is suitable
for preprocessing the whole data set of our experiment.

5   Point Clouds Merging Process

The process of point clouds merging will be performed in four stages. All of them
are illustrated for clarity in Fig. 11 using a matchbox.

The sub-picture Fig. 11(a) depicts the general startup position of two neigh-
bouring point clouds. Subsequent sub-pictures depicts following four stages of
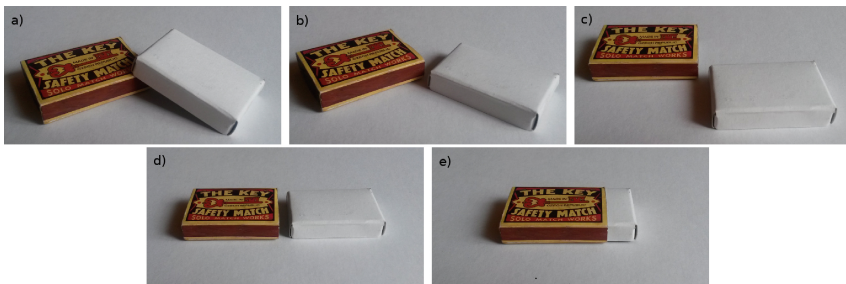merging process:

Fig. 11. Illustration of point clouds merging stages on the process of closing a matchbox

1. Fig. 11(b) - the horizon leveling.
2. Fig. 11(c) - the direction alignment.
3. Fig. 11(d) - the centering.
4. Fig. 11(e) - the insertion.

The main advantage of proposed 3D LRF design will be used in following process. The advantage is the possibility of scanner to measure the whole environment including a space even above the scanner. Just the ceiling is the most visible part of the environment. It is flat, without obstacle and in long term in the unchanging state. Thus the ceiling will be in center of attention in the most stages of point clouds merging process.

5.1   Horizon Leveling

The measuring device does not stay during the measurement always in ideal horizontal position. The floor has minor irregularities which cause small inclination of the device. The tilt in some cases is greater than few degrees.

In this stage the horizon will be balanced according to the ceiling plane. The segment of ceiling is in all point clouds the largest segment of all. It covers the area of approximately 10 to 16 square meters. So the informative value of these segments is very high and it can be used in a simple way. The normal vector of the ceiling segment can be easily computed in centroid of the segment ceiling. This normal vector will consequently define a new direction of the vertical axis z.

The balancing of the whole point cloud will be realized by rotations around axes x and y. The necessary angles of rotations can be easy computed from the angle between the normal vector $\boldsymbol{n}$ and the axis z. All points in every point cloud will be transformed by following formula:

$$\boldsymbol{p}_{r,s} = R_y(\alpha_y) \cdot R_x(\alpha_x) \cdot \boldsymbol{p}_{r,s}. \tag{3}$$

5.2   Direction Alignment

In this stage the ceiling segment will be reused again. The direction of the point cloud will be based on left and right edges of the ceiling segment. The top view of this situation is depicted in Fig. 12.

The measuring device is in general position in the origin of the coordinate system O. The virtual lines, parallel with the axis y, are regularly marked at distances d on both sides of the axis y. On the left and right side these lines leaves ceiling segment in edge points L_i and R_j. The connecting lines of these points creates a set of vectors $\boldsymbol{l_i}$ and $\boldsymbol{r_j}$. Now these vectors will be used to determine the direction in two steps: estimation of direction and improving its accuracy.

The estimation of the direction will be computed separately for the left and right side of the ceiling segment using the following formulas:
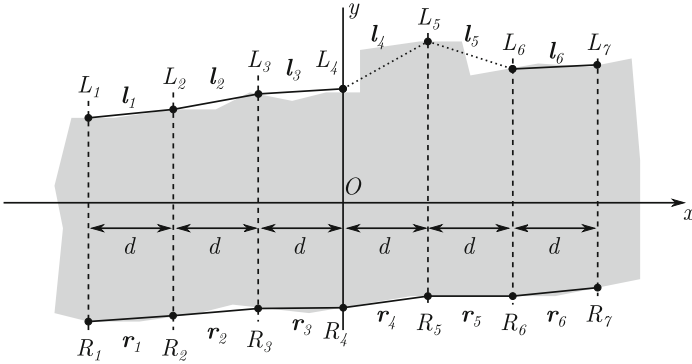
Fig. 12. The segment representing the ceiling used for the direction alignment

$$l = \frac{\sum_{i=1}^{M} l_i}{\sum_{i=1}^{M} |l_i|}, \quad r = \frac{\sum_{j=1}^{N} r_j}{\sum_{j=1}^{N} |r_j|}, \quad (4)$$

where M and N is a number of left and right vectors.

The estimation will be used for a simple filtering. Vectors from the left and right set, which divert from the vector l and r more than the given angular threshold, will be removed. In this experiment this threshold is set to $10°$. This simple filtering removes bounces and niches in wall from both ceiling edges. Removed vectors are marked in Fig. 12 by dotted line.

The number of vectors will be reduced by this filtering to number M' and N' and final direction d is computed from all remaining vectors by the formula:

$$d = \frac{\sum_{i=1}^{M'} l_i + \sum_{j=1}^{N'} r_j}{\sum_{i=1}^{M'} |l_i| + \sum_{j=1}^{N'} |r_j|}. \quad (5)$$

Now it remains only to calculate the angle α_z between the vector d and the axis x. Every point cloud will be rotated around the axis z by this angle:

$$p_{r,s} = R_z(\alpha_z) \cdot p_{r,s}. \quad (6)$$

5.3   Centering

This is the first stage where a pair of neighbouring point clouds will be used. Furthermore, the odometric value of the shift between two measurements is needed. As was mentioned above, this distance is approximately 1 meter. This distance will be marked as d_o.

In this stage, two neighbouring point clouds will be placed side by side, on its joined axes x at the distance d_o. This situation is partially visible in Fig. 13. The illustration depicts only left edges of two neighbouring point clouds. Elements
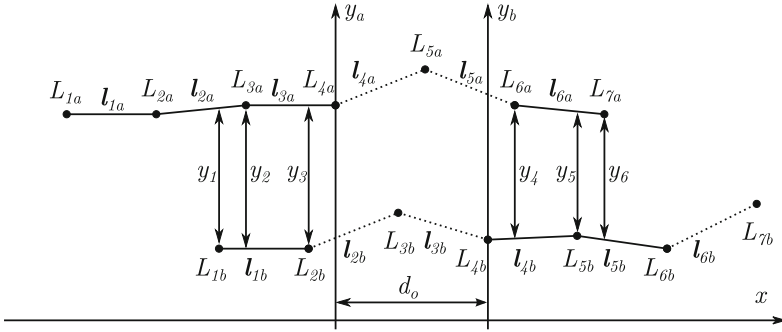
Fig. 13. The left edges of two point clouds

of the first point cloud are marked with additional index a and elements of the second point cloud with index b.

Now it is necessary to compute distances between all points L_i of cloud a and edges l_j of cloud b and vice versa. The same procedure is applied for right edges of both point clouds. The distances between points and removed edges are not computed.

The result is a set of distances y_k and the number of distances can be marked as K. The average value of all y_k can be computed by the following formula:

$$y_a = \frac{\sum_{k=1}^{K} y_k}{K}. \tag{7}$$

The average distance will be used for translation of the whole point cloud b by distance y_a:

$$\boldsymbol{p}_{r,s} = T_y(-y_a) \cdot \boldsymbol{p}_{r,s}. \tag{8}$$

5.4   Insertion

The last stage of point clouds merging will use odometric distance d_o and corresponding front and back segments of neighbouring point clouds.

The estimation of the insertion is based on the distance d_o. The improvement of accuracy is performed by measuring the distance of corresponding front and back segments. The pair of segments corresponds when the distance of their centroids is less than 20 % of d_o. The distances of centroids are only computed in the direction of the axis x. These distances are averaged in the similar way as in previous stage. The result leads to more accurate distance d_o'. This distance d_o' can be used for the translation of all points of point cloud b:

$$\boldsymbol{p}_{r,s} = T_x(d_o') \cdot \boldsymbol{p}_{r,s}. \tag{9}$$

Now it is possible to merge all points from point cloud a and b into single point cloud.

6   Merged Point Cloud

The merging process described in the previous section must be used for all point clouds in the data set as well as for all neighbouring point clouds. The result of this process is a single point cloud that merges 25 point clouds. The result is visible in Fig. 14. This point cloud is composed of 5.4×10^6 points. Thus in this figure it seems solid, not as a point cloud.

Fig. 14. The top view and the side view of the whole corridor - 25 merged point clouds
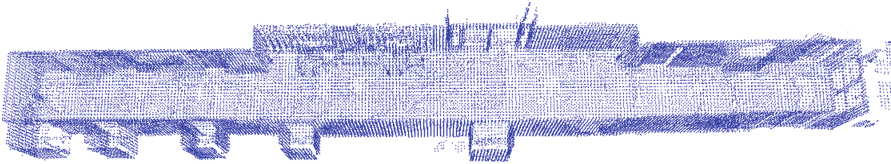

Fig. 15. The merged point cloud after voxelization (100 mm)

Figure 14 shows that the merging process did not create an exact straight object. The result is a little bit curved. But this result was achieved by using point clouds only without any additional position or incline sensors.

For further usage it is necessary to reduce the amount of points of the merged point cloud. For the reduction of points a *voxelization* algorithm is used. It this process the whole space is divided into small cubes, for this experiment the cube size is 100 mm, and all points inside every cube is reduced to a single point. Obtained results directly correspond to the Octomaps used for the robot navigation [7]. The result of this process is visible in Fig. 15. This point cloud is composed of only 46×10^3 points. It is less than 1 % of the original merged point cloud size.

The segmentation algorithm can be repeatedly applied on the merged point cloud with reduced number of points.

The result of this segmentation is visible in Fig. 16. There is clearly visible that the segmentation recognizes the ceiling very well as the single (red) segment.
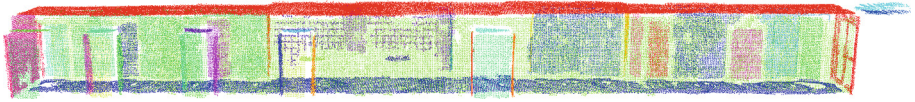
Fig. 16. Side view of the resulting segmentation of the reduced point cloud (Color figure online)
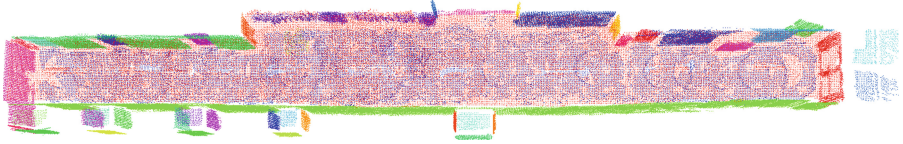

Fig. 17. Top view of the resulting segmentation of the reduced point cloud (Color figure online)

The same result is for the floor which is depicted as a single (blue) segment. In Fig. 17 it is also visible that the segmentation can very well recognize niches and doors. The resulting

Based on this result, it is possible to say that the process of point clouds merging was successful.

7   Conclusion

The paper introduces a novel concept of using cheap 2D LRF as a 3D scanning device. In contrast to more advanced devices, additional procedures of calibration and point cloud merging is necessary, however the final results are comparable with the more expensive devices. This article focuses on the process of merging point clouds and describes it in more details. Special properties of indoor measurements are used as a basis for establishing merging parameters. Merged point clouds represent whole environment and can be used after the segmentation as a source for the robot navigation and environment mapping. The experiments showed that for the successful merging only a small part of raw measurements is needed (in the described example less than 1 % of acquired points were used).

References

1. Bosse, M., Zlot, R., Flick, P.: Zebedee: design of a spring-mounted 3-D range sensor with application to mobile mapping. IEEE Trans. Robot. **28**(5), 1104–1119 (2012)
2. Rusu, R.B.: Semantic 3D object maps for everyday manipulation in human living environments. Disertation, Institut für Informatik der Technischen Universität München (2009)
3. Alexandre, L.A.: 3D descriptors for object, category recognition: a comparative evaluation. In: Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, p. 7 (2012)

4. Filipe, S., Alexandre, L.A.: A comparative evaluation of 3D keypoint detectorsin a RGB-D object dataset. In: 2014 International Conference on Computer Vision Theory and Applications (VISAPP), vol. 1, pp. 476–483. IEEE (2014)
5. Xiao, J., Adler, B., Zhang, H.: 3D point cloud registration based on planar surfaces. In: 2012 IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp. 40–45. IEEE, September 2012
6. Xiao, J., Adler, B., Zhang, J., Zhang, H.: Planar segment based three-dimensional point cloud registration in outdoor environments. J. Field Robot. **30**(4), 552–582 (2013)
7. Wurm, K.M., Hornung, A., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: a probabilistic, flexible, and compact 3D map representation for robotic systems. In: Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation, vol. 2, May 2010
8. Krumnikl, M., Olivka, P.: PWM nonlinearity reduction in microstepping unit firmware. Przegld Elektrotech. **88**(3a), 232–236 (2012)
9. Olivka, P., Krumnikl, M., Moravec, P., Seidl, D.: Calibration of short range 2D laser range finder for 3D SLAM usage. J. Sens. **501**, 3715129 (2016)
10. I Heart Robotics, More Hokuyo 3D Laser Scanner Images, October 2015. http://www.iheartrobotics.com/2009/06/more-hokuyo-3d-laser-scanner-images.html
11. Point Cloud Library. http://www.pointclouds.org
12. Velodyne LiDAR, HDL-64E. http://www.velodynelidar.com/hdl-64e.html