The development of point cloud registration algorithm using geometrical features of the multi-sensor 3D scanner

Vladimir Gromov and Pavel Kustarev

ITMO University, Faculty of Software Engineering and Computer Systems,
Saint-Petersburg, Russia
gromov@vladimir.one
kustarev@corp.ifmo.ru
ifmo.ru

Abstract. The aim of the research is a 3D point matching which uses models from different sources to union them in the general coordinate system to create a high-accuracy scene from the 3D scanner. There are several images from the stereo camera (dual optical camera) at the device output to construct a 3D high-resolution model with 360 degrees horizontal field of view. There is a high-accuracy point cloud from the LIDAR module (laser rangefinder) to correct the full field view from the stereo camera. The research process is called point cloud registration and will help to create an output 3D representation with advantages of both sources. The presented algorithm has low computational complexity and,in the future, will be implemented on the scanner embedded system. The results of the data processing are presented in this paper.

Keywords: Point cloud registration, Data fusion, 3d scanner, Stereo camera, LIDAR, OpenCV, SteroBM, StereoSGBM

1 Introduction

The object of the research is the multi-sensor mobile 3D scanner with 360 degrees range, which uses the stereo camera and LIDAR module. Calculations are made on the embedded heterogeneous computing system with GPU NVidia Tegra TK1 and FPGA Xilinx Zynq-7000 soc [1]. Figure 1 shows the target 3D scanner.

The resolution of a 360 degrees model from the stereo camera is up to 10 millions colored points on the target system. However, the representation has low accuracy and includes unrecognized parts of the scene with solid-color objects without shadows.

The point cloud from the laser rangefinder Sweep V1 has high accuracy up to 1 centimeter for 40 meters distance [2]. However, there are few numbers of points which approximately equals 10 thousand from the system. Figure 2 shows the scene object which has been recognized by stereo camera and LIDAR.

Both of clouds in the Cartesian coordinate system are turned versus each other, they have different angles of view and shifted origin points.

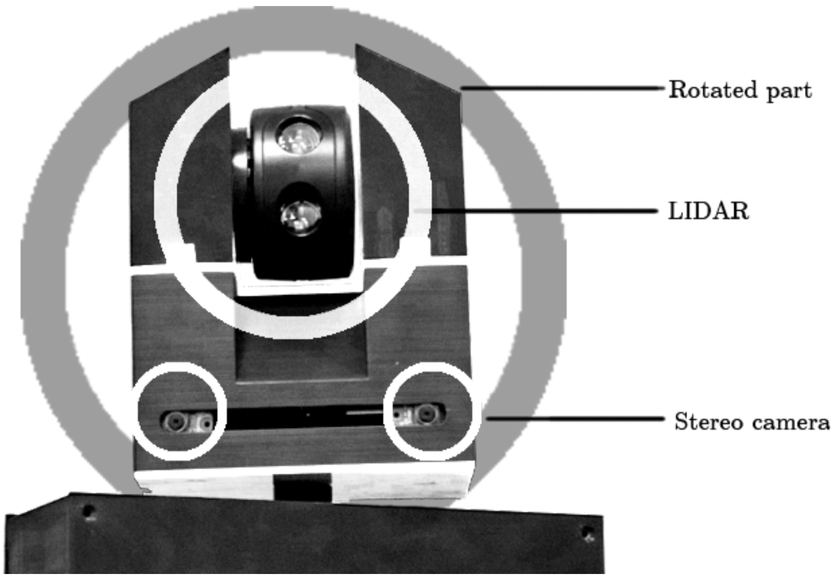

Fig. 1. The 3D scanner with LIDAR and the stereo camera module on the rotated part

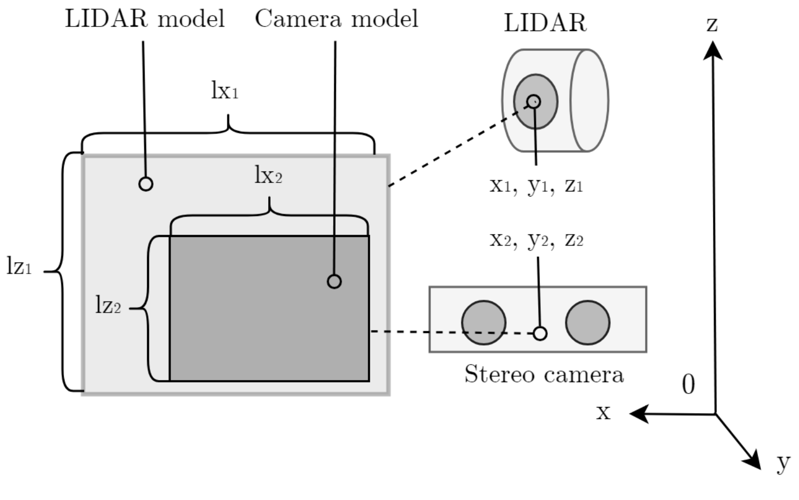

Fig. 2. The 3D model representation from LIDAR and the stereo camera

Point cloud registration is used to build a new representation, which contains source models from the stereo camera and LIDAR and will be used to correct the output scene. The general approach to the point cloud registration is using key points for shifting and rotating input models to the scene construction. It uses geometric features of source clouds and makes a high-quality matching with data that has not any distortions [3] [4].

In practice, there are many distortions in the stereo camera model, and we cannot use this method without significant modifications [5].

Therefore, we chose an approach, which uses some scanner geometry features in lieu model key points. We decided to divide the data fusion task into two subtasks.

The first task is the 3D scene construction from the scanner with 360 degrees range. The second one is the data fusion of combined stereo camera model with LIDAR point cloud. Both of subtasks could be solved by key point and geometric approaches. The decomposition allows to understand the task clearly and to get the estimated result, which could help to evaluate the matching quality.

2 Materials and methods

2.1 Stereo camera model

The stereo camera is consisted of two calibrated coaxial oriented camera modules, which give a couple of calibrated images for disparity map calculation to create the 3D scene. We use StereoBM and StereoSGBM methods for disparity map calculation with the use the default calibration module from OpenCV library [6]. Each of calibrated images for disparity map calculation are shown on fig. 3.

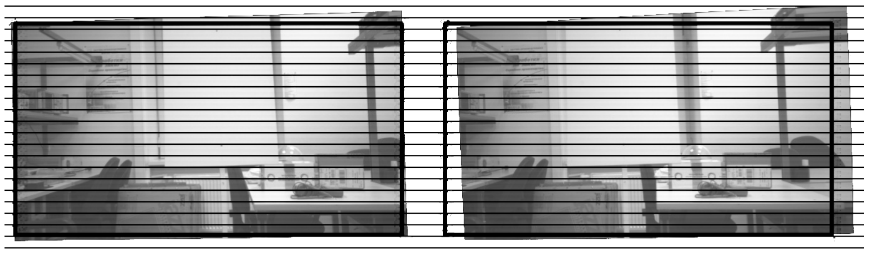

Fig. 3. Calibrated images from the stereo camera

The disparity map is a two-dimensional array, which keeps the distance between two horizontal identical points in both pictures. We use the equation below for disparity map conversion to 3D model [7].

$$\begin{cases} y &= f \times b/disp \\ x &= xl \times y/f \\ z &= yl \times y/f \end{cases} \tag{1}$$

Where
x, y, z - point coordinates in the 3D model in mm.
xl, yl - point coordinates in the disparity map in pixels.
f - focal length of the camera in mm.
disp - disparity value in pixels.
b - baseline, the distance between two camera modules in mm.

Figure 4 shows the calculated 3D model based on the disparity map.

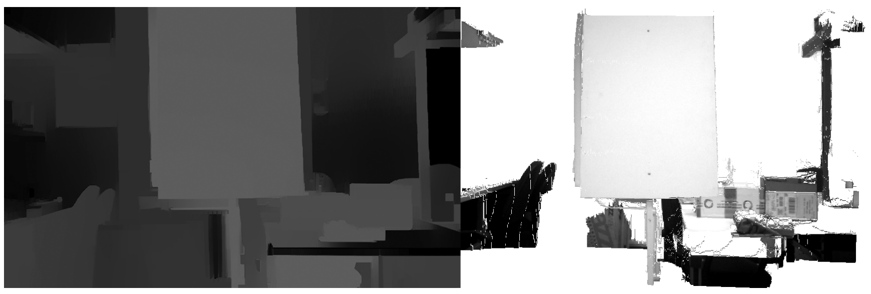

Fig. 4. The 3D model based on the disparity map

2.2 LIDAR model

Laser rangefinder Sweep V1 creates a 2D scanning in the horizontal axis of rotation. This module has been installed on the scanner rotating part and does vertical scanning. After finishing, the 2D vertical scanner part is turned for a few degrees to perform the next 2D scanning. Figure 5 shows the point cloud from LIDAR.

2.3 Horizontal angle view calculation

Stereo camera has been installed on the horizontal rotated part of the scanner to create a 360-degrees range model. The scanner turns by $\Delta\ \phi$ view angle from the previous state to take a couple of photos. Figure 6 shows the top view of the scanner which makes new a couple of images every $\Delta\ \phi$ angle.

We use equation 2 to calculate $\Delta\ \phi$ view angle in degrees.

$$\Delta\phi = \alpha \times binning \times dispRes/maxRes \tag{2}$$

Where
α - camera horizontal view angle in degrees.
dispres - horizontal resolution of the disparity map.
maxRes - maximum horizontal resolution of the camera matrix.
binning - camera binning factor, which depends on the camera operation mode.

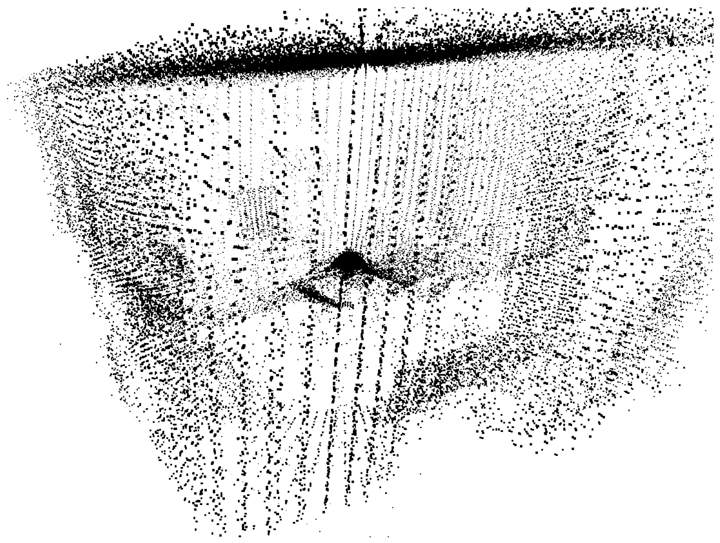

Fig. 5. The 3D point cloud from *LIDAR*

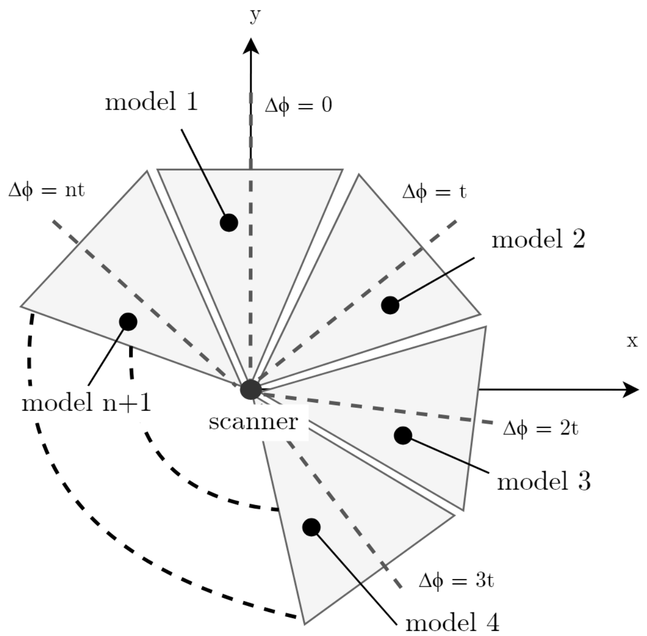

Fig. 6. The rotation scheme

Values *α, maxRes and binning* are got from camera specifications [8] [10], value *dispRes* is got from the OpenCV function getValidDisparityROI [11].

The equation 3 presents the calculation of $\Delta\ \phi$ degrees for the system.

$$\Delta\phi = 62,2 \times 2 \times 1121/3280 = 42,51 \tag{3}$$

2.4 360-degree model construction from the stereo camera

For 360-degrees scene construction we turn every model by $\Delta\ \phi$ degrees from the previous state. For rotation, we obtain spherical coordinates from the cartesian coordinate system using the ordinary equation [11] and then add $\Delta\ \phi$ angle to every ϕ spherical coordinates.

The rotation equation 4 in the spherical coordinate system is used to construct the scene.

$$\begin{cases} r &= r_{nModel} \\ \theta &= \theta_{nModel} \\ \phi &= \phi_{nModel} + (n-1) \times \Delta\phi \end{cases} \tag{4}$$

Where n $\in$ *[1...N], N* is the quantity of sources models, which equals (5)

$$360/42,5 = 8,4 = 9. \tag{5}$$

This approach with using some geometric features (view angle and $\Delta\ \phi$ rotation angle) has advantages over the key points method.

Firstly, the deformation of input models does not affect to 360 degrees model construction. Secondly, fewer models are used for construction because it is not necessary to cross them with each other. Finally, there is a simple implementation, which does not require high-performance hardware resources. The method also has a disadvantage. Angle errors and vibrations in the mechanical part of the device will influence on the 360 degrees range stereo model and won't be founded using this method.

2.5 Stereo camera model

Figure 7 shows the top view of sensors shifting versus each other in the scanner.

We are using several features of the 3D scanner to match models. As you can see, the stereo camera and laser rangefinder have a different height (Z coordinates are not equal), but other axis variables are equal.

The first step is the moving the point cloud from LIDAR to few centimeters down along the Z-axis. In our system, we have to move the scene to 5 cm down.

In addition, there is an angle shift. Laser rangefinder is located above the stereo camera thus zero horizontal angle of the LIDAR model equals a half of the stereo camera view angle.

The second step of matching is subtracting from every horizontal angle of the LIDAR model a half of view angle.

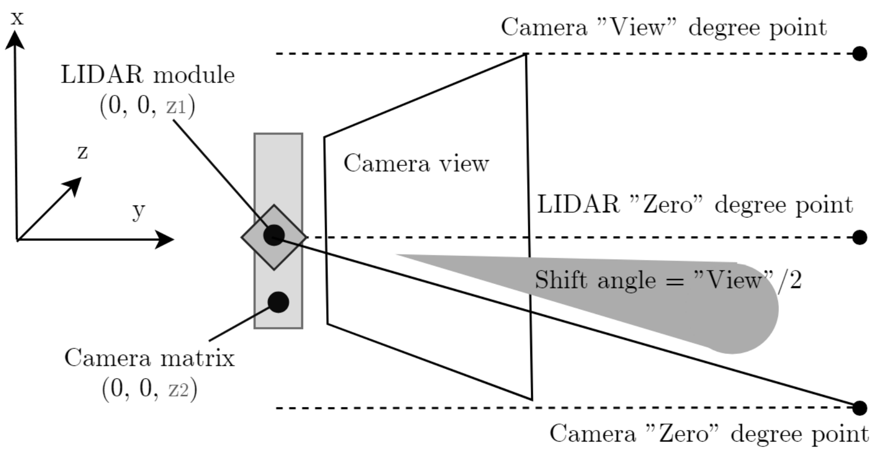

Fig. 7. The top of view of the stereo camera and LIDAR shifting

Steps below provide matching of LIDAR and stereo camera point clouds.
1. Offset the laser rangefinder model down in the Cartesian coordinate system(6).

$$\begin{cases} x_{shift} &= x_{lidar} \\ y_{shift} &= y_{lidar} \\ z_{shift} &= z_{lidar} - n \end{cases} \tag{6}$$

2. Convert the shifted scene to the spherical coordinate system.
3. Rotate the model in spherical coordinates system (7).

$$\begin{cases} r &= r_{shift} \\ \theta &= \theta_{shift} \\ \phi &= \phi_{shift} - \Delta\phi/2 \end{cases} \tag{7}$$

For improving calculation performance we shift LIDAR cloud because there are fewer points than in the stereo camera model.

3 Results

3.1 Developed software

We developed the software in C++ which creates a union point cloud from the stereo camera and laser rangefinder using geometrical method described above. Both representations are kept in different organized structures to implement fast calculations. Figure 8 shows the LIDAR point cloud structure.

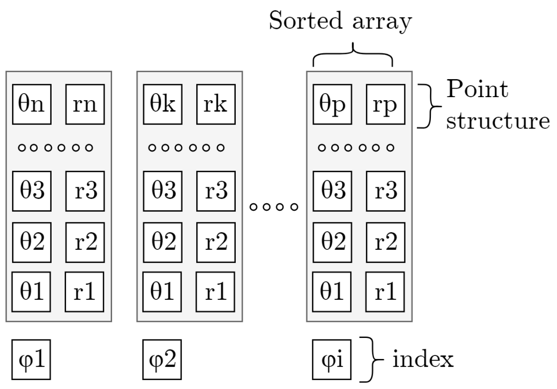

Fig. 8. The LIDAR point cloud structure

The main feature of the LIDAR model is that every point of the 2D range in a plane has equal ϕ coordinate. There are one-dimensional arrays with point structure which consists of θ angle and radius. We can get an array through ϕ indexing. This structure is used to keep fewer values and to get simple indexation. In addition, for rotating we change only the index value of the array. So, there is no need to change every point.

The stereo camera model has other features. Unfortunately, we cannot assume that every point in a line has equal ϕ in column case and θ in row space. Equation [11] means that ϕ and θ also depend on disparity value in every cell, so both of coordinates are not equal in the same line. However, at every point, we can get all neighbors from left, right, top and down parts like on 2D-image because disparity map is a matrix structure. Therefore, we decided to save the matrix indexation, but we kept radius, ϕ and θ values in each cell as well. Figure 9 shows the data structure of points array for models from the stereo camera.

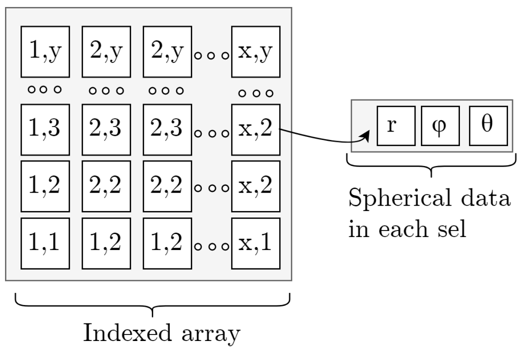

Fig. 9. The keeping structure for the 360 degrees view model

The array height size is equal to the disparity map array height. The array width size is equal to width of the disparity map array, which is multiplicated by the number of source models to make a 360-degrees view scene. Both of structures will help us to correct output model in the future because there is a simple indexation to get the neighbor point from each sell.

3.2 Experiment

We scanned the next scene to evaluate the quality of the model combination:

- 9 colored clouds from the stereo camera with 42 degrees view angle;
- LIDAR point cloud where every 3 degrees a 500 points plane has been built.

Figure 10 shows the example of the first 3D model from the stereo camera.

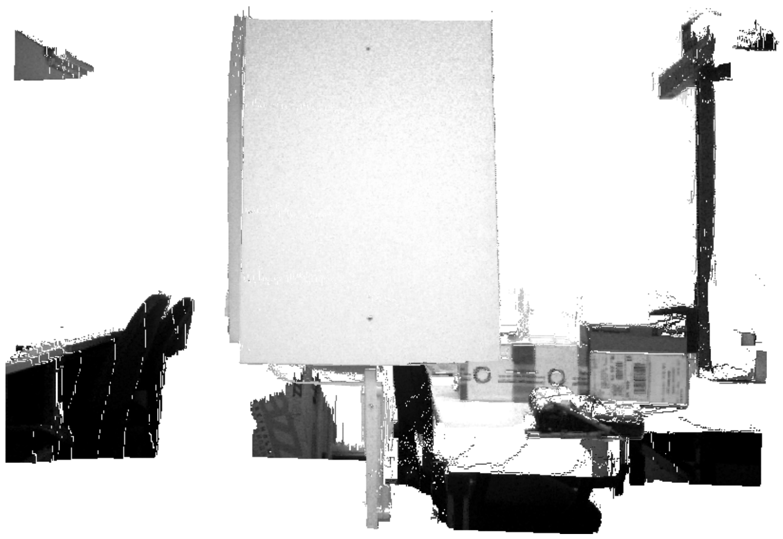

Fig. 10. The model from the stereo camera

Figure 11 shows the projection of the part LIDAR point cloud to scene image. Figure 12 shows 360-degrees view model from the stereo camera.

The quality of input cameras clouds is low, but the combining process has matched well. Figure 13 shows the combining of the first and the ninth model.

They are crossing because there is a remainder after division 360 degrees by 42. As you can see, there is a crossing in a box image between the first and the ninth model. The picture on the box looks like it has been matched correctly at the intersection. It shows that the angle view and calculations are correct. Figure 14 shows the combined 3D model from the stereo camera and the laser rangefinder.

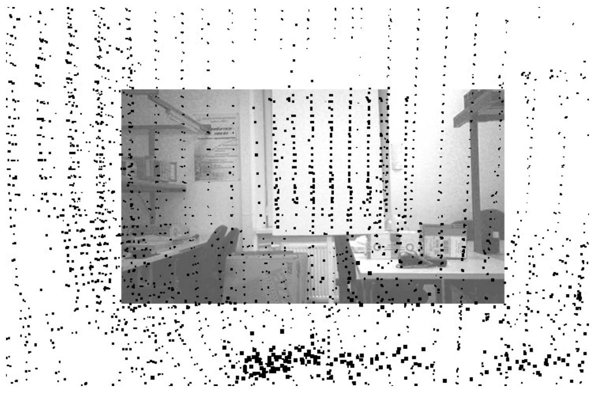

Fig. 11. The projection of a laser rangefinder point cloud to the stereo camera image

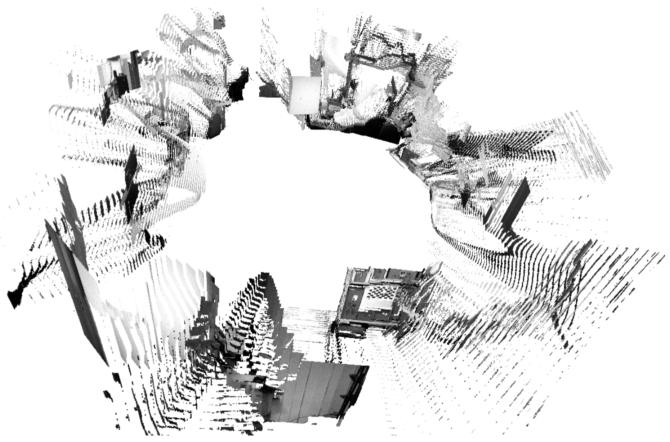

Fig. 12. The combined 3D model from the stereo camera

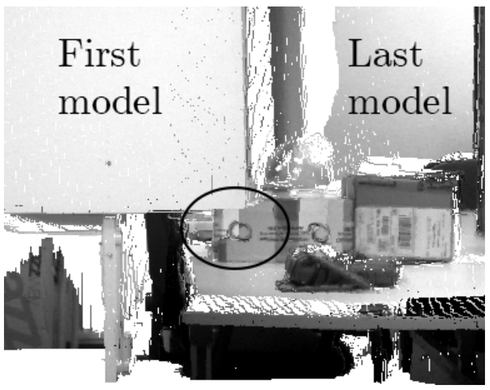

Fig. 13. The combining of the first and the last model

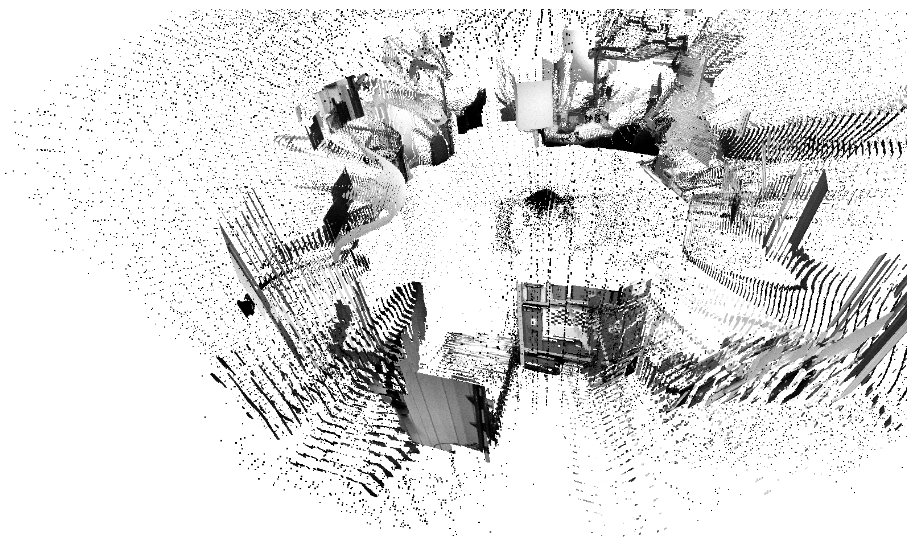

Fig. 14. The combined 3D model from the stereo camera and LIDAR

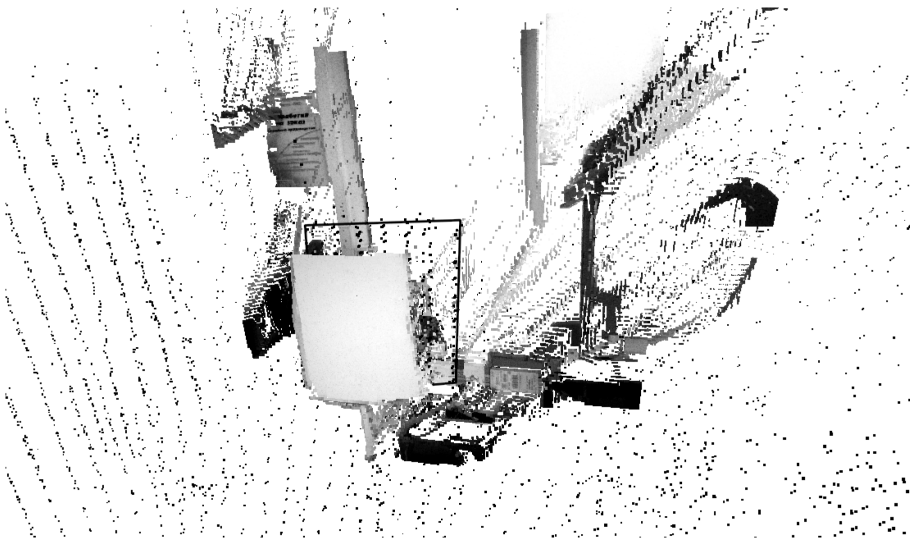

Fig. 15. The partial combining between the stereo and LIDAR models

Figure 15 shows the partial combining between the first stereo camera model and the LIDAR point cloud to show the matching quality.

The part of the whiteboard in the laser rangefinder and stereo camera clouds is highlighted in Fig. 15. There is a 10 cm distance detection error in the stereo camera model which is caused by some distance errors in the input source scene. The single threaded application calculates the result scene for 18 seconds without reading and writing operations on CPU Intel Core I7 4770. The representation cloud consists of 5 500 000 points. The accuracy depends on the real precision of rotation movement of the scanner module.

4 Discussion

As a result of this research, the algorithm and software have been developed and tested with a real data from the 3D scanner. The future work is the construction of a laboratory scene with some metrical objects to evaluate algorithm accuracy. The other way is a developing a method for point cloud correcting using presented data structures and scanner characteristics. In the future, we are going to improve the application performance using a multi-threaded implementation.

References

1. Bikovsky S., Kustarev P., Didin E., Gromov V., Dranitsa A.: Mobile systems for mediumrange 3D. Control Engineering, Russia, P 34–37 (2018)
2. Scanse, Sweep V1. User manual and technical specifications, P 1 (2017)
3. Pomerleau F, Colas F, Siegwart R. :A Review of Point Cloud Registration Algorithms for Mobile Robotics, (2015)
4. The PCL Registration API, `http://pointclouds.org/documentation/tutorials/registration_api.php`
5. Rusu R.: Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments. Dissertation, P 199 (2009)
6. Camera Calibration and 3D Reconstruction, `https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html`
7. Scharstein D. : View synthesis using stereo vision, P 10 (1999)
8. Camera specification , Raspberry pi offical site, `https://www.raspberrypi.org/documentation/hardware/camera`
9. API Camera Calibration and 3D Reconstruction, OpenCV, official site, `https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html`
10. Camera hardware, ReadTheDocs, `https://picamera.readthedocs.io/en/release-1.13/fov.html`
11. Spherical coordinate system, Wikipedia, `https://en.wikipedia.org/wiki/Spherical_coordinate_system`