Research Article

International Journal of Advanced Computer Research, Vol 10(48)
ISSN (Print): 2249-7277 ISSN (Online): 2277-7970
http://dx.doi.org/10.19101/IJACR.2020.1048031

Emphasis of LiDAR data fusion using iterative closest point and ICP registration

Shashidhara H S[*] and Naveen Kumar S
Information Science and Engineering, Ramaiah Institute of Technology, Bangalore, India

Received: 30-March-2020; Revised: 20-May-2020; Accepted: 25-May-2020

Abstract

Light detection and ranging (LiDAR) is an optical remote-detecting technique that utilizes laser light to densely sample the earth's surface, delivering exceptionally exact x, y, z estimations. Lieder, essentially utilized in airborne laser mapping applications, is developing as a financially savvy option in contrast to conventional looking over strategies, for example, Photogrammetry. Spatially sorted post prepared LiDAR information is known as point cloud data (PCD) and LiDAR produces these point cloud datasets in mass. The underlying point mists are vast accumulations of 3D height focuses, which incorporate x, y, z and intensity. Data can be collected from multiple LiDARs simultaneously, which are placed adjacent to each other like one on each of the headlights in front of the car. Each LiDAR produces its own PCD. A method is required to combine the PCD from participating LiDARs for further analysis. In this paper, a novel approach for combining the data from multiple LiDAR is proposed, which involves obtaining inliers and outliers. The efficient algorithms such as conversion algorithm, Iterative Closest Point and ICP registration are used in the process. The conversion algorithm is applied to point cloud data to render the mesh representation of the actual image from PCD. Outliers are separated using distance algorithms such as Euclidean and are discarded. Inlier data are treated with Iterative Closet Algorithm (ICP) to generate a matrix of points. Finally, ICP registration has been applied to consecutive data frames from adjacent LiDARs and combined PCD is retrieved resulting in the fusion of PCDs. The LiDAR data fusion has various applications in the field of autonomous driving.

Keywords

LiDAR, PCD, Conversion algorithm, Iterative closest point, ICP registration.

1.Introduction

LiDAR is an examining approach that estimates the distance to an object with pulsed laser light by illuminating the object and computing the reflected pulses with a sensor. LiDAR is a remote detecting innovation, which makes use of pulses from a laser to gather estimations which would then be able to be utilized to make 3D models and maps of objects and environments. LiDAR works similar to Radar and Sonar yet utilizes light waves from a laser rather than radio or sound waves. A LiDAR framework works by calculating distance after sending light wave and computing to what extent it takes for the light to hit an item or surface and reflect back to the scanner. The distance is then determined employing the velocity of light. The way the LiDAR measures the surroundings is by knowing the Time of Flight. A firmware present in LiDAR analyses and measures the data when the laser is fired. An optical lens collects the data from the objects in the vicinity. Even though millions of light pulses are fired, the system can process only around 100 pulses which contain information.

Data created from LiDAR incorporates the crude point information as well as refined derivatives such as contours and surfaces. Point data blends the elevation values, yet in addition intensity values, classification values, and certain other point attribute. LiDAR data are accessible from various open and private elements, and can be acquired openly or at an expense. In either case, it is vital to choose information that will support the predetermined purpose. There are many stipulations to contemplate when obtaining LiDAR data, and the level of flexibility extended by various sources fluctuates significantly.

The LiDAR fusion requires point cloud data which will be available from open data sources such as kitty

[*] Author for correspondence

dataset. Point cloud gathers data points characterized by a given coordinate system. A point cloud may delineate the state of some genuine obstacle in a 3D framework. Point clouds are utilized to construct 3D networks and other different models used in 3D demonstration for a variety of fields, including therapeutic imaging, design, 3D printing, manufacturing, 3D gaming and various computer-generated applications. The data are taken in PCD format and used for further processing.

A methodology to combine PCDs from multiple LiDARs was required which would result in the smooth transition from one PCD to another. This improves the decision to be taken by automated system about next step. Both ICP and ICP registration algorithms were used on image data earlier with some results. As PCD is similar to pixel data of images, there is an attempt to utilize the same on PCD.

Objectives of the work can be summarized as:

i. Use conversion algorithm to obtain the mesh diagram.
ii. Apply outlier algorithm to remove unwanted points which may increase computational time.
iii. Use iterative closest point algorithm to decide point data to be considered by comparing adjacent frames.
iv. Use ICP Registration algorithm to obtain combined PCD of adjacent frames.

1.1Literature survey

Wei et al. [1] have suggested a novel method where vehicles use LiDAR and monochrome camera to prevent vehicles from entering into a restricted space which is a high valued private asset. They have used beacons to mark these areas which are identified by LiDAR but suffer from false positives like workers safety vests.

Hajri and Rahal [2] stated that a standard station wagon with two high-resolution contrast and grey scale cam recorders are prepared for this purpose. The exact ground truth is given by a Velodyne laser scanner and a GPS confinement framework. Datasets are caught by driving around the fair-size city of Karlsruhe, in on highways and in rustic regions. Up to 15 vehicles and 30 people are detectable per image. Other than giving all information in raw configuration, benchmarks are extracted for each task.

Vosselman [3] paper titled advanced point cloud processing shows some extraordinary models on how point cloud divisions can be used to normally isolate Geo-information. The paper focusses on the extraction of man-made items in the urban condition. The models, consolidate the treatment of point fogs got through airborne, adaptable similarly as terrestrial laser scanners.

Sugimoto et al. [4] presents examples of advancements to address 3D point fogs and its use cases are investigated. The usage of 3D point fogs has been growing, and there is a need for gainful depiction of 3D point fogs. Work coding after surface fitting has been one of the major approaches for point cloud weight, and weight exploiting the relationship after improving the primary point cloud is moreover transforming into an important procedure starting late.

To increase the precision of perception in automated vehicles, heterogeneous sensors like LiDAR, radar, ultrasound sensors and cameras are being employed. They capture multidimensional data and in different formats and different alignments. De Silva et al. [5] have proposed a novel approach based on geometric modelling and Gaussian Process regression method.

A method to process large amount of LiDAR data is proposed by in their work Lyu et al. [6]. They have designed an FPGA system for Real-time road segmentation using convolution networks. It was possible to achieve good response time but at the cost of on-chip memory. The observed limitation was false positive in some situations, but the notable limitation was the requirement for fusion of LiDAR data to remove the false positives.

A deep learning approach for road detection using LiDAR data was proposed by Caltagirone et al. [7]. A set of statistics are chosen to generate top-view point cloud images. They try to prove that top-view perspective 2D distribution of points contain sufficient information for road detection. However, selected statistics might not result in the most optimal method for road detection.

Xiao et al. [8] have proposed a novel hybrid conditional random field (CRF) algorithm to fuse data from monocular camera and multi-layer LiDAR. It can be used to reduce ambiguities in detecting road surfaces. But the method does not give good average precision values. The reason is average precision is designed for evaluating probabilistic predictions but the method proposed in this work is based on binary prediction.

One of the methods, UGrid – Fused, in the paper by Wulff et al. [9] is used to create multi-dimensional occupation grid representation of integrated data from camera and LiDAR. This grid will be the input for fully convolutional network (FCN) which gives more accurate surface detection. The paper does not handle fusion of multiple LiDAR data.

LiDAR histogram method by Chen et al. [10] efficiently combines detection of traversable road regions, obstacles and water hazards into single framework. The 3d traversable road plane in front of the vehicle can be projected as straight-line segment and positive and negative obstacles can be drawn above and below the line. This converts the entire problem of differentiating road and obstacles into simple linear classification. Since the LiDAR data has been converted to point clouds, this method cannot generate LiDAR imagery

2.Materials and methods

The initial dataset in is the text format which is from Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) dataset. One of the extensively used dataset is KITTI dataset which is a computer vision benchmark established in 2012. The KITTI dataset takes advantage of autonomous driving platform to create novel testing certifiable computer vision benchmarks. The interested areas are such as optical flow, stereo, 3D object detection, visual odometry, and 3D tracking.

The dataset comprises the following information:

- Uunsynced and Unrectified raw as well as synced and rectified processed grayscale stereo sequences stored in png format each of 0.5 Megapixels
- Uunsynced and Unrectified raw as well as synced and rectified processed colour stereo sequences stored in png format each of 0.5 Megapixels
- 3D point clouds of 100k points per frame, stored as binary float matrix. (This is the important data upon which the proposed work is built). Each tuple in the dataset comprises of x, y, z and intensity values. Number of tuples in point cloud depends upon the image.
- 3D GPS/IMU data where location, speed, acceleration, meta information are stored as text file
- Calibration in terms of Camera, Camera-to-GPS/IMU, Camera-to-Velodyne stored as text file
- 3D object tracklet labels of cars, trucks, trams, pedestrians, cyclists, stored as xml file

Unsynced and unrectified refers to the raw input frames where images are distorted and the frame indices do not correspond, while synced and rectified refers to the processed data where images have been rectified and undistorted and where the data frame numbers correspond across all sensor streams.

Data is taken from kitty dataset which is in the format of text file. The dataset is freely available from the open source which can be downloaded. The text data is converted to PCD by using conversion algorithm which involves conversion code. The .txt data is transformed to .pcd after being processed by the conversion algorithm. The converted. pcd data is best suitable for the further implementation process.

2.1Conversion algorithm

Point clouds are used to build maps and estimate vehicle trajectory from LiDAR data. Conversion Algorithm is used to convert the LiDAR data which is in .txt format to 3D mesh representation. The steps are as mentioned below:

i.Registering point clouds – consists of creating pose graphs, down sampling a 3D pint cloud, geometric transformation etc.

ii.Fitting point clouds to geometric models – involves fitting plane to 3d point cloud, estimating normal for point cloud, fitting spheres to 3d point cloud etc.

iii.Reading and writing point clouds – consists of reading 3d point cloud from pcd file, writing 3d point cloud to pcd file

iv.Visualizing point clouds – involves plotting 3d point clouds and visualizing streaming 3d point cloud

v.Storing point cloud – create object which stores 3d point cloud, finding points within a region of interest etc.

The 3D rendered by conversion algorithm consists of two types of information viz. relevant for further algorithm and not relevant to next step. For example, the road reflection, trees besides the road are irrelevant. The other vehicles and objects like animals and obstacles are relevant for further processing. For iterative closet point algorithm, which is the next step, to work we need to input only relevant data. A small outlier algorithm is applied to output of conversion algorithm which removes outliers such as road reflection and trees and passes only inliers such as animals and obstacles to ICP.

2.2Iterative closest point

In this technique, point cloud density geometric characteristics are used to search the corresponding connection between point clouds resulting in properly registered points. However, it is difficult to get all the objects point cloud data from the same point of perspective because 3D scanning instrument has a field of view restriction. It is necessary to integrate the part of the surface PCD obtained from different angles to get overall PCD of the object. Hence, the next step of registering where point cloud's 3D coordinates are matched and overlapped properly at distinct angles.

ICP algorithm involves the general steps as below (*Figure 1*):

- For each point (from the entire arrangement of vertices for the most part alluded to as thick or a determination of sets of vertices from each model) in the source point cloud, Match the nearest point in the reference point cloud (or a chose set).
- Determine the consolidation of translation and rotation by making use of a root mean square point to point distance metric minimization method which is the best adjust each source point to its match found in the past advance. This progress may likewise include weighting points and dismissing anomalies earlier arrangement.
- Remodel the source points enhancing the acquired change.
- Then the process is iterated.

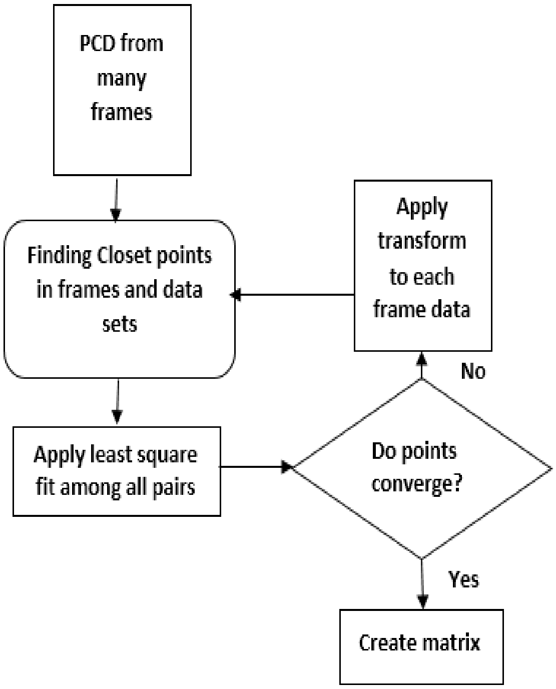

PCD from many frames
Finding Closet points in frames and data sets
Apply transform to each frame data
Apply least square fit among all pairs
Do points converge?
No
Yes
Create matrix

Figure 1 Iterative closet point algorithm

2.3ICP registration

The two kinds of ICP registration are explained here.

A. Point to point ICP

Universally, it includes two stages:

1. Find the matrix set K= {(p, q)} from the target point cloud P, and transform the source point cloud Q with the present transformation matrix T.
2. Update the T transformation by diminishing the defined objective function E (T) through the K match set.

Distinct alternatives use distinct functions of objective E (T):

Initially Point to point algorithm using the following objective function is utilized in the ICP registration:

$$E\ (T) = \sum_{(p,q)\epsilon k} (p - Tq)$$

B. Point to plane ICP

This algorithm makes use of another function of objective as below:

$$E\ (T) = \sum_{(p,q)\epsilon k} ((p - Tq)\,.\,n_p)^2$$

Where n_p is the normal of point p. This technique has a quicker conjunction rapidity than the technique point to point ICP.

For the point cloud data, the inliers and outliers are extracted by using sections of inliers and outliers code. For each data the separate files of inliers and outliers are fetched. The outliers are ignored while the inliers are concentrated for next step. The inliers are treated by Iterative closest Point algorithm which outputs matrix points. The matrix is of size 4*4. The next move is to employ the ICP registration to the matrix points. As this step outputs overlapped PCD data, two consecutive point cloud data are considered for ICP registration to get the finest overlapping. The perfect overlap is achieved by performing the all the above processes.

The block diagram of same is depicted in *Figure 2*. In pairwise registration, interesting points are detected which best represent the scene in both data sets. A feature descriptor is computed at each key point. A set of correspondences are estimated on the basis of similarities between positions and features along with XYZ coordinates. Correspondences having negative values are rejected assumed to be noisy. Final set of coordinates are calculated from good correspondences which are required for motion transformation estimation.

2.4Implementation environment

Algorithm was run on a system with 2.4 Hz speed and 4GB RAM with Ubuntu 14.04 OS. The algorithm is implemented using Python 3D and Open3D.

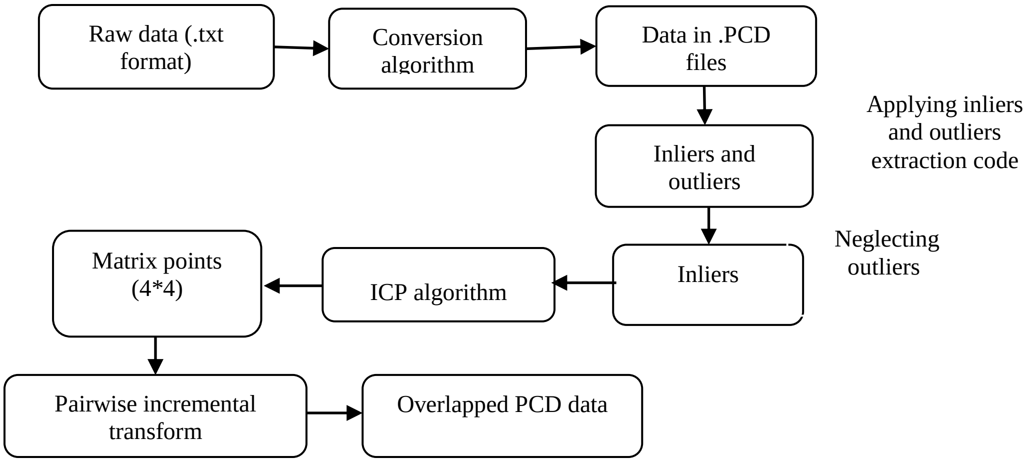


Figure 2 Proposed flow of LiDAR data fusion

The Outputs presents the fused LiDAR data which is differentiated by colours. The fusion is legitimate when consecutive frames are taken from the dataset.

3.Results

The proposed work is implemented and the results are obtained as shown in the following screenshots. The RGB image sample, whose .pcd format is used as lidar data is shown in *Figure 3*. The image data file is downloaded in the text format which is shown below in *Figure 4*. Each image data includes the four data points such as x, y, z and intensity. The same is depicted in *Figure 5*. When the conversion algorithm is applied to text data, the point cloud data is accessed for the same image data, which is presented in *Figure 6* and *Figure 7*. A small section of inlier and outlier code is applied to the above achieved point cloud data, which results in separating inliers and outliers from the data. Separate files are created for inliers and outliers point cloud data that are shown in the *Figures 8* and 9. As the outlier data is no longer needed, it is neglected and inlier data is treated with Iterative closest point (ICP) algorithm resulting in matrix of size 4×4. The matrix points are displayed in *Figure 10*. Finally, the ICP registration is applied to two consecutive data frames and the overlapped image in point cloud data format is retrieved. *Figure 11* and *Figure 12* show the results. The blue and orange colour represents the two different data frames which are overlapped on one another.

Comparison between point-point and point-plane ICP registration methods:

Point-plane ICP registration fails to meet the correct course of action by becoming asymptotic after 6 iterations. Point-point algorithm however improves as iterations progress giving almost nil residual error in merging of two PCDs (*Table 1*).

Table 1 Comparison between point-point and point to pane ICP registration methodologies

		Relative residual error (% Per Dataset)	
		Point-point ICP	**Point-plane ICP**
Iterations	2	0.02	∞
	4	0.009	∞
	6	0.006	0.015
	8	0.005	0.015
	10	0.004	0.015
	12	0.002	0.015
	14	0	0.015

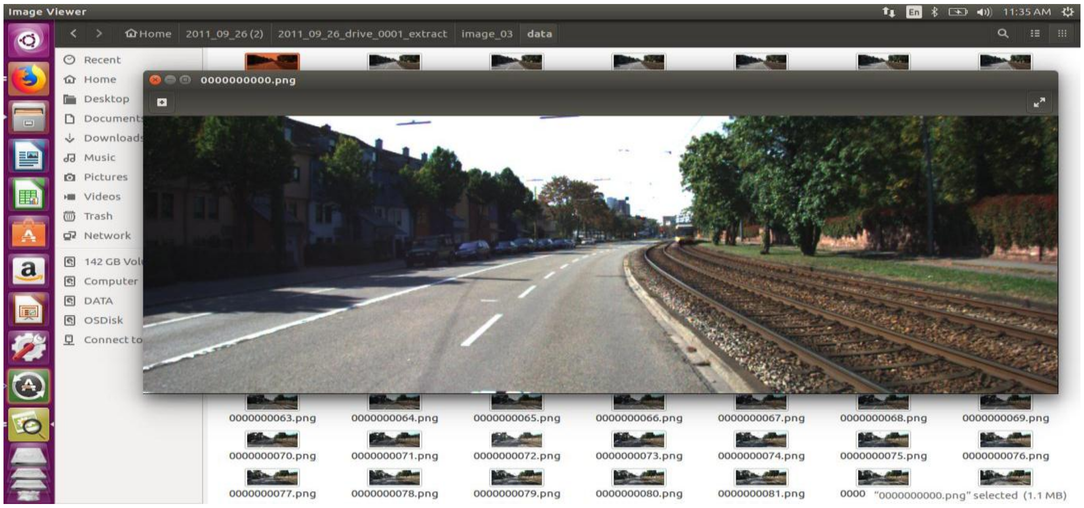


Figure 3 RGB image

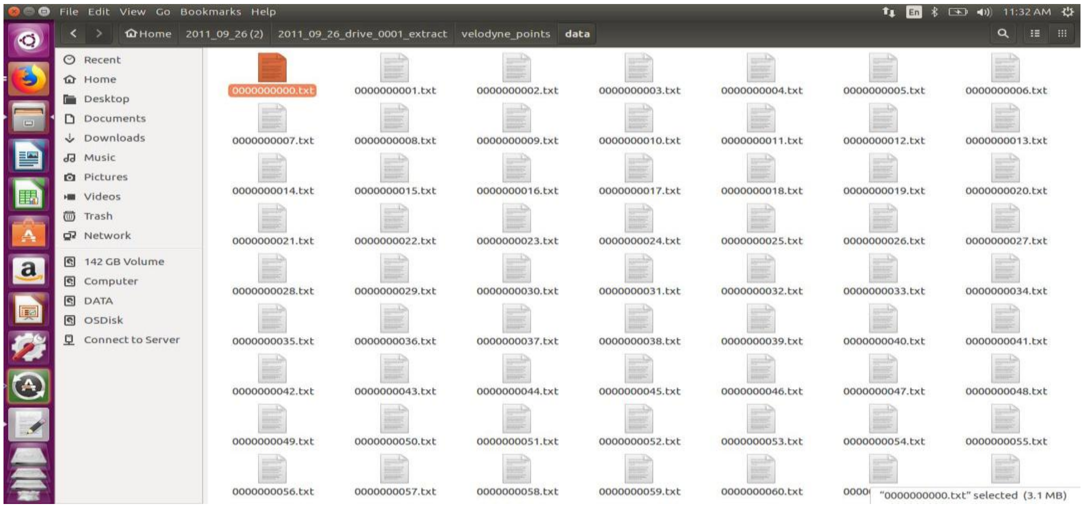
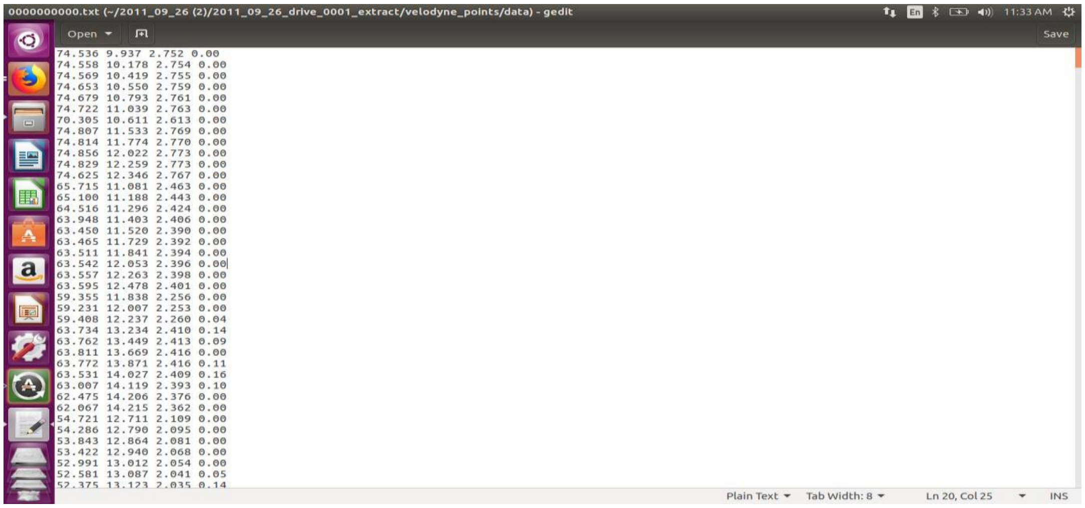
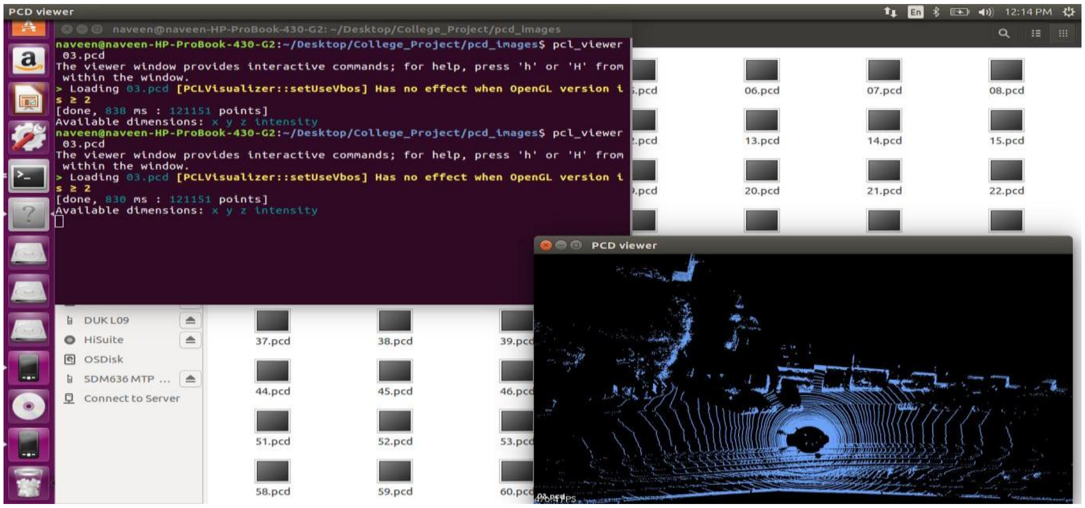


Figure 4 Image data file in text format


0000000000.txt (~/2011_09_26 (2)/2011_09_26_drive_0001_extract/velodyne_points/data) - gedit
11:33 AM
Open
Save
Plain Text
Tab Width: 8
Ln 20, Col 25
INS

```
74.536 9.937 2.752 0.00
74.558 10.178 2.754 0.00
74.569 10.419 2.755 0.00
74.653 10.550 2.759 0.00
74.679 10.793 2.761 0.00
74.722 11.039 2.763 0.00
70.305 10.611 2.613 0.00
74.807 11.533 2.769 0.00
74.814 11.774 2.770 0.00
74.856 12.022 2.773 0.00
74.829 12.259 2.773 0.00
74.625 12.346 2.767 0.00
65.715 11.081 2.463 0.00
65.100 11.188 2.443 0.00
64.516 11.296 2.424 0.00
63.948 11.403 2.406 0.00
63.450 11.520 2.390 0.00
63.465 11.729 2.392 0.00
63.511 11.841 2.394 0.00
63.542 12.053 2.396 0.00
63.557 12.263 2.398 0.00
63.595 12.478 2.401 0.00
59.355 11.838 2.256 0.00
59.231 12.007 2.253 0.00
59.408 12.237 2.260 0.04
63.734 13.234 2.410 0.14
63.762 13.449 2.413 0.09
63.811 13.669 2.416 0.00
63.772 13.871 2.416 0.11
63.531 14.027 2.409 0.16
63.007 14.119 2.393 0.10
62.475 14.206 2.376 0.00
62.067 14.215 2.362 0.00
54.721 12.711 2.109 0.00
54.286 12.790 2.095 0.00
53.843 12.864 2.081 0.00
53.422 12.940 2.068 0.00
52.991 13.012 2.054 0.00
52.581 13.087 2.041 0.05
52.375 13.123 2.035 0.14
```

Figure 5 x, y, and z and intensity data points



```
naveen@naveen-HP-ProBook-430-G2:~/Desktop/College_Project/pcd_images$ pcl_viewer
 03.pcd
The viewer window provides interactive commands; for help, press 'h' or 'H' from
 within the window.
> Loading 03.pcd [PCLVisualizer::setUseVbos] Has no effect when OpenGL version i
s ≥ 2
[done, 838 ms : 121151 points]
Available dimensions: x y z intensity
naveen@naveen-HP-ProBook-430-G2:~/Desktop/College_Project/pcd_images$ pcl_viewer
 03.pcd
The viewer window provides interactive commands; for help, press 'h' or 'H' from
 within the window.
> Loading 03.pcd [PCLVisualizer::setUseVbos] Has no effect when OpenGL version i
s ≥ 2
[done, 830 ms : 121151 points]
Available dimensions: x y z intensity
```

Figure 6 Point cloud data

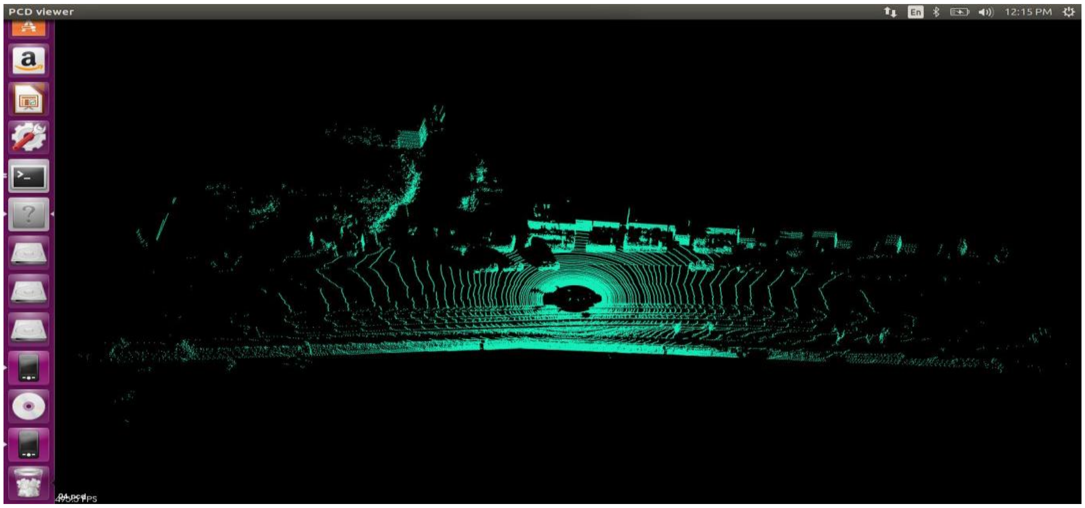


Figure 7 PCD format of image data

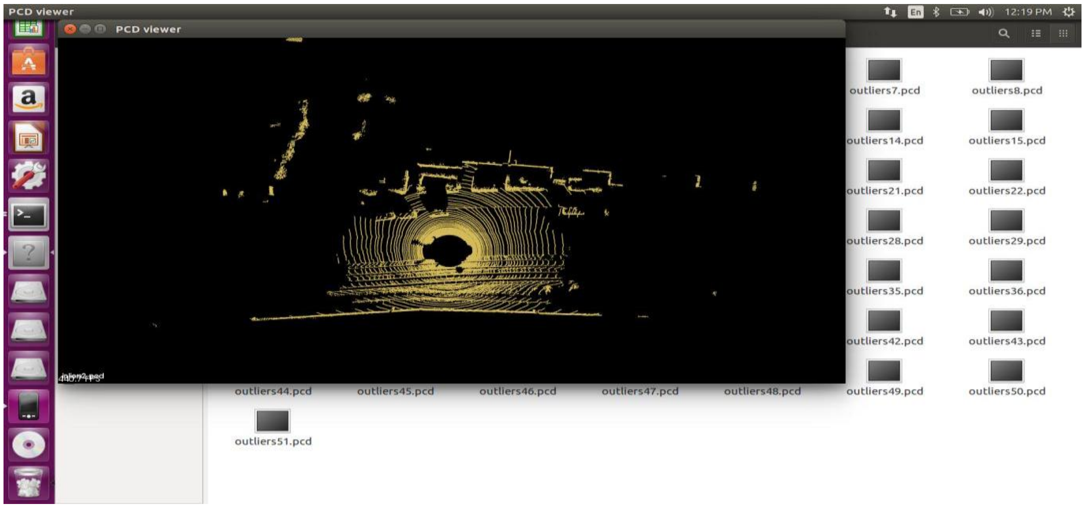


Figure 8 Inlier data

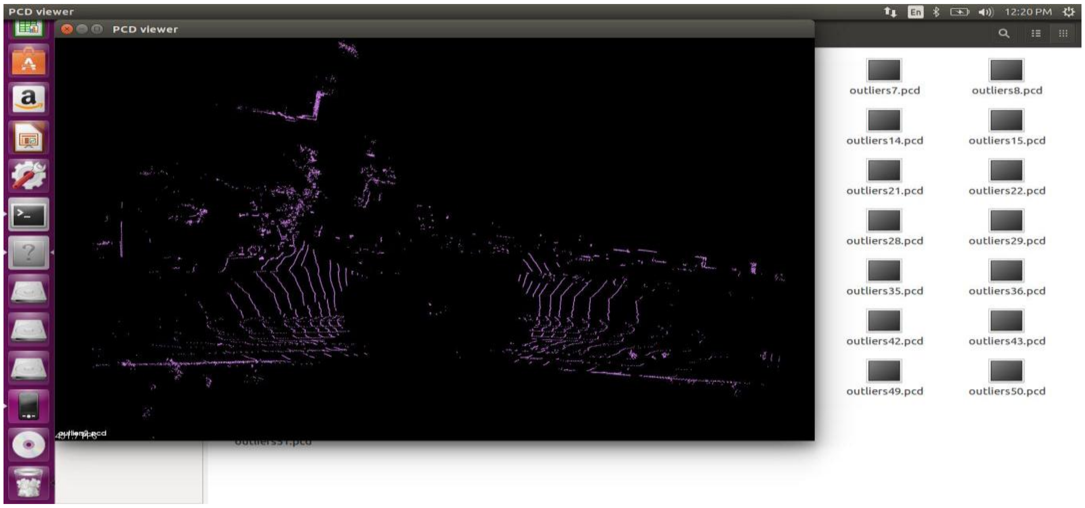
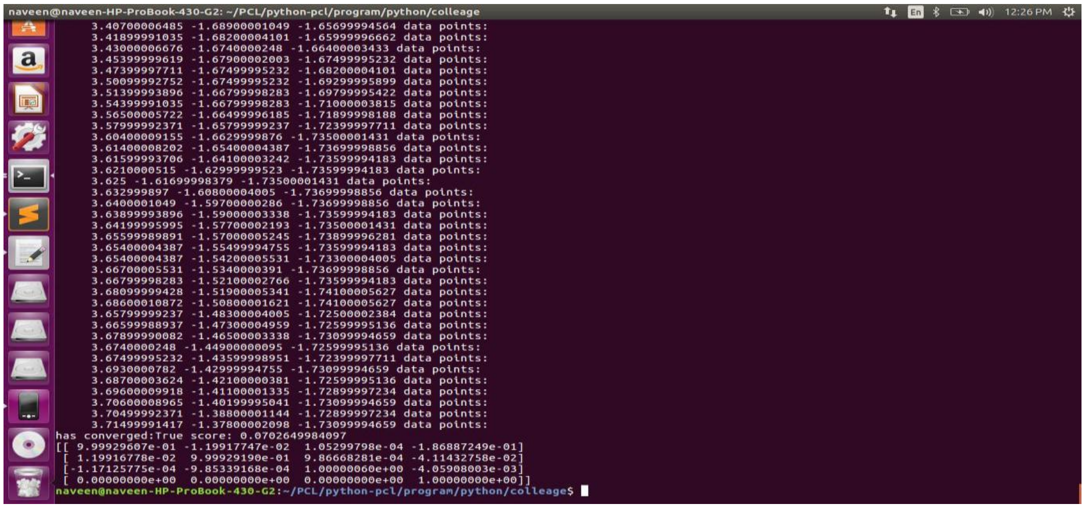


Figure 9 Outlier data


naveen@naveen-HP-ProBook-430-G2: ~/PCL/python-pcl/program/python/colleage
12:26 PM

```
        3.40700006485 -1.68900001049 -1.65699994564 data points:
        3.41899991035 -1.68200004101 -1.65999996662 data points:
        3.43000006676 -1.6740000248 -1.66400003433 data points:
        3.45399999619 -1.67900002003 -1.67499995232 data points:
        3.47399997711 -1.67499995232 -1.68200004101 data points:
        3.50099992752 -1.67499995232 -1.69299995899 data points:
        3.51399993896 -1.66799998283 -1.69799995422 data points:
        3.54399991035 -1.66799998283 -1.71000003815 data points:
        3.56500005722 -1.66499996185 -1.71899998188 data points:
        3.57999992371 -1.65799999237 -1.72399997711 data points:
        3.60400009155 -1.6629999876 -1.73500001431 data points:
        3.61400008202 -1.65400004387 -1.73699998856 data points:
        3.61599993706 -1.64100003242 -1.73599994183 data points:
        3.6210000515 -1.62999999523 -1.73599994183 data points:
        3.625 -1.61699998379 -1.73500001431 data points:
        3.632999897 -1.60800004005 -1.73699998856 data points:
        3.6400001049 -1.59700000286 -1.73699998856 data points:
        3.63899993896 -1.59000003338 -1.73599994183 data points:
        3.64199995995 -1.57700002193 -1.73500001431 data points:
        3.65599989891 -1.57000005245 -1.73899996281 data points:
        3.65400004387 -1.55499994755 -1.73599994183 data points:
        3.65400004387 -1.54200005531 -1.73300004005 data points:
        3.66700005531 -1.5340000391 -1.73699998856 data points:
        3.66799998283 -1.52100002766 -1.73599994183 data points:
        3.68099999428 -1.51900005341 -1.74100005627 data points:
        3.68600010872 -1.50800001621 -1.74100005627 data points:
        3.65799999237 -1.48300004005 -1.72500002384 data points:
        3.66599988937 -1.47300004959 -1.72599995136 data points:
        3.67899990082 -1.46500003338 -1.73099994659 data points:
        3.6740000248 -1.44900000095 -1.72599995136 data points:
        3.67499995232 -1.43599998951 -1.72399997711 data points:
        3.6930000782 -1.42999994755 -1.73099994659 data points:
        3.68700003624 -1.42100000381 -1.72599995136 data points:
        3.69600009918 -1.41100001335 -1.72899997234 data points:
        3.70600008965 -1.40199995041 -1.73099994659 data points:
        3.70499992371 -1.38800001144 -1.72899997234 data points:
        3.71499991417 -1.37800002098 -1.73099994659 data points:
has converged:True score: 0.0702649984097
[[ 9.99929607e-01 -1.19917747e-02  1.05299798e-04 -1.86887249e-01]
 [ 1.19916778e-02  9.99929190e-01  9.86668281e-04 -4.11432758e-02]
 [-1.17125775e-04 -9.85339168e-04  1.00000060e+00 -4.05908003e-03]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00]]
naveen@naveen-HP-ProBook-430-G2:~/PCL/python-pcl/program/python/colleage$
```

Figure 10 Matrix points

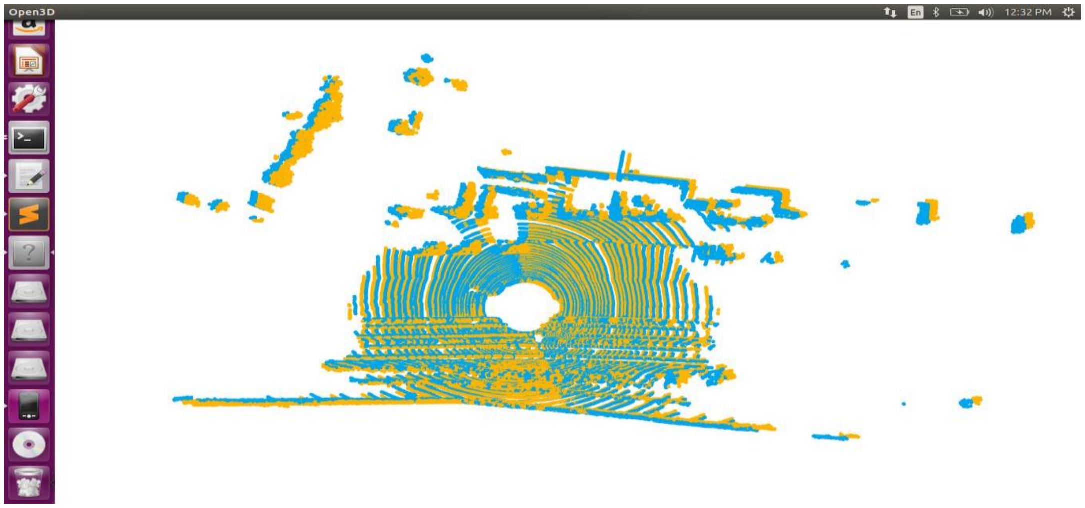


Figure 11 The overlapped point cloud data

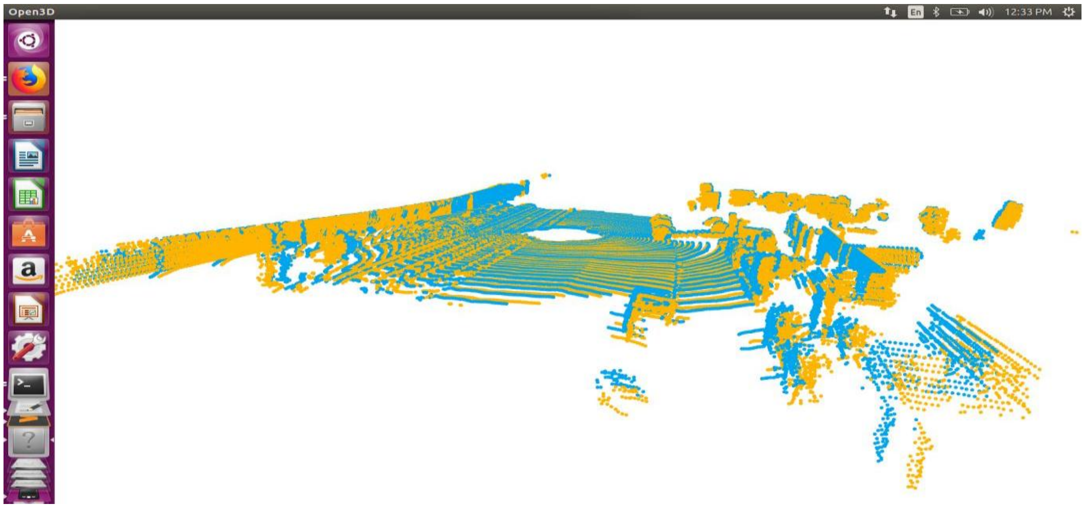


Figure 12 Two consecutive overlapped data frames

4.Discussion

Image data which is in .pcd format is downloaded in .txt form and conversion algorithm is used to generate mesh structure. Outlier algorithm is applied to separate it from inliers and outliers are discarded. inlier data is treated with Iterative closest point (ICP) algorithm resulting in matrix of size 4*4. Finally, the ICP registration is applied to two consecutive data frames and the overlapped image in point cloud data format is retrieved.

Conversion algorithm, ICP and ICP registration were earlier used independently to convert pcd data to 3d representation, finding inliers and finding the difference in pcd data in two different frames. In this work, there is an attempt to combine all the three algorithms to get a continuous 3d rendering of consecutive frames. This feature will be very useful in autonomous vehicles which need to process information to make decisions about moving ahead smoothly. As LiDAR data is light weight compared to camera images, the processing is also much quicker.

5.Conclusion and future work

The point cloud data can be employed to depict, in addition to other things, a lot amount of estimation data created by 3D scanning devices. Point clouds are used for many purposes, inclusive of constructing 3D CAD models for fabricated parts, for quality inspection and metrology, and for a multiple view of representation, rendering, animation, and mass customization operations, also used for reverse engineering of CAD information from produced models. Hence the input data in any format must be primarily converted to .pcd format in order to get expected outcome.

The codes employed for conversion of text data to point cloud data as well as extracting inliers and outliers are simple and will directly output the requisite results. With the help of which the appropriate proceeding steps can be undertaken. The ICP algorithm which is applied for getting nearest match points is the convenient approach for obtaining matrix points. By which twelve matrix points gathered in the form of 4×4. This form of matrix can be processed by ICP registration which produces registration of the point cloud data.

The future work of this proposed methodology for LiDAR data fusion can be comprised of:

- Saving the overlapped PCD data. The resulted PCD data is converted to .vtk format. And the preserved data is displayed in visualization tool kit using VTK file format.
- After being processed to VTK format, the purview of the data can be shown
- And further, it can also be upgraded to video data.
- The exemplified work can also be replaced with enhanced versions of the used algorithms to get even better refinement in the results.
- The future work needs some more analysis to be done to save the outcomes and use it for further implementation.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] Wei P, Cagle L, Reza T, Ball J, Gafford J. LiDAR and camera detection fusion in a real-time industrial multi-sensor collision avoidance system. Electronics. 2018; 7(6):1-32.

[2] Hajri H, Rahal MC. Real time lidar and radar high-level fusion for obstacle detection and tracking with evaluation on a ground truth. arXiv preprint arXiv:1807.11264. 2018.

[3] Vosselman G. Advanced point cloud processing. In Photogrammetric Week 2009.

[4] Sugimoto K, Cohen RA, Tian D, Vetro A. Trends in efficient representation of 3D point clouds. In asia-pacific signal and information processing association annual summit and conference (APSIPA ASC) 2017 (pp. 364-9). IEEE.

[5] De Silva V, Roche J, Kondoz A. Fusion of LiDAR and camera sensor data for environment sensing in driverless vehicles. arXiv 2018. arXiv preprint arXiv:1710.06230.

[6] Lyu Y, Bai L, Huang X. Real-time road segmentation using lidar data processing on an fpga. In international symposium on circuits and systems 2018 (pp. 1-5). IEEE.

[7] Caltagirone L, Scheidegger S, Svensson L, Wahde M. Fast LIDAR-based road detection using fully convolutional neural networks. In intelligent vehicles symposium (iv) 2017 (pp. 1019-24). IEEE.

[8] Xiao L, Wang R, Dai B, Fang Y, Liu D, Wu T. Hybrid conditional random field based camera-LIDAR fusion for road detection. Information Sciences. 2018; 432:543-58.

[9] Wulff F, Schäufele B, Sawade O, Becker D, Henke B, Radusch I. Early fusion of camera and lidar for robust road detection based on U-Net FCN. In intelligent vehicles symposium (IV) 2018 (pp. 1426-31). IEEE.

[10] Chen L, Yang J, Kong H. Lidar-histogram for fast road and obstacle detection. In international conference on robotics and automation 2017 (pp. 1343-8). IEEE.

Shashidhara H S is from Bangalore, Karnataka, India. He has M.Tech. degree in Computer Science and Engineering and currently pursuing Ph. D as part time research scholar under Jawaharlal Nehru Technological University, Hyderabad. His areas of interest include Bioinformatics, IoT and Software Design Patterns. He is currently working as faculty in the department of Information Science and Engineering, Ramaiah Institute of Technology, Bangalore.
Email: shashi@msrit.edu

Naveen Kumar S is from Hunsur, Karnataka, India. He has M.Tech. degree in Software Engineering. He is currently working as a Software Developer in KPIT Technology Ltd. Bangalore. His areas of interest include Machine Learning, Deep Learning and Image Processing.
Email: naveensrinivas43@gmail.com