A Mobile System for 3D Indoor Mapping Using LiDAR and Panoramic Camera

WEIMIN WANG[1] KENJI YAMAKAWA[1] KEI HIROI[2] KATSUHIKO KAJI[1]
NOBUO KAWAGUCHI[2]

Abstract: Limited by the poor vertical resolution of multi-laser LiDAR sensors like Velodyne HDL-32E, the sparsity of 3D points increases drastically in vertical direction as the distance from Velodyne increases. Interpolating in the point cloud is a method that often used to solve this problem. In this paper, we propose a mobile system consist of a Velodyne LiDAR sensor and a panoramic camera(we utilize a Ladybug3 panoramic camera here) for indoor mapping. We call it Velobug(Velodyne and Ladybug) system for practical purpose. Velobug can be used to obtain denser points clouds by deflecting Velodyne and rotating it around plumb line. Also with the Ladybug3 panoramic camera under Velodyne, we can get a colored point cloud by corresponding each 3D point to a pixel on the panoramic image acquired from Ladybug3 after the calibration of Velobug. Generalized Iterative Closet Point(Generalized-ICP) is adopted in frame-to-frame registration of the point clouds for indoor mapping.

1. Introduction

For indoor spatial information services such as indoor location service or indoor disaster management, IndoorGML[1] has been proposed in [2]. It is a trend that maps are changing from conventional 2D maps to 3D maps not only for outdoor environment but also for indoor environment. 3D mapping is an essential procedure of IndoorGML. 3D indoor reconstruction and mapping with RGB-D sensors like Kinect has been proposed in [3]. However, effective range of these RGB-D sensors are generally less than 5 meters(Table 1). With the effective range and the viewing angle in Table 1, the area that a frame of RGB-D sensors cover is less than $10m^2$. This would cost lots of time to acquire enough 3D data for indoor mapping in large indoor environment like station, shopping mall or museum. Even more, accumulative error caused by frequent registration may lead to the mis-registration of the whole indoor environment.

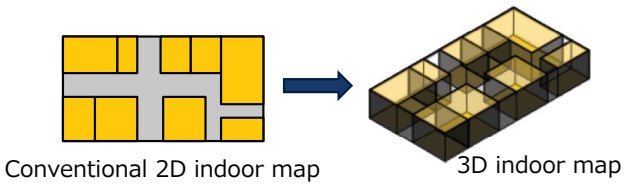


Fig. 1: 2D indoor map is evolving into 3D

For many advantages like wider viewing angle, farther effective range (Table 2) and insensitivity to visible light,

Table 1: Specification of main RGB-D sensors

Product	Viewing Angle	Effective Range	RGB
Microsoft Kinect for XBOX 360	$57° \times 43°$	$1.2m - 3.5m$	Yes
Panasonic D-IMager EKL3105	$60° \times 44°$	$1.2m - 5.0m$	No
MESA Imaging SR4000	$69° \times 56°$	$0.3m - 5.0m$	No
ASUS Xtion PRO LIVE	$58° \times 45°$	$0.8m - 3.5m$	Yes
Occipital Structure Sensor	$58° \times 45°$	$0.4m - 3.5m$	No
Microsoft Kinect for Windows v2	$70° \times 60°$	$0.5m - 4.5m$	Yes

Table 2: Specification of popular LiDAR sensors

Product	Viewing Angle	Effective Range
Velodyne HDL-64E	$360° \times 26.8°$	$< 120m$
Velodyne HDL-32E	$360° \times 41.3°$	$1m - 70m$
HOKUYO 3D-LIDAR	$210° \times 40°$	$0.3m - 50m$
HOKUYO UTM-30LX	$270° \times 0.25°$	$0.1 - 30m$

LiDAR sensors have been used in many fields such as autonomous system, robot vision and SLAM(Simultaneous Localization and Mapping). To overcome the shortage in effective range of RGB-D sensors, we think of introducing LiDAR to 3D indoor reconstruction and mapping. Time cost will be substantially reduced benefited from wide measurement range of LiDAR. Furthermore, mis-registration caused by accumulative error is excepted to be improved as area covered by one frame increases. However, LiDAR measures distance by calculating the time difference between transmission and deflection of laser beam, and thereby no color information can be obtained.

In this paper, we propose a mobile system for indoor reconstruction and mapping. We utilize a Velodyne HDL-32E LiDAR sensor to acquire 3D data more effectively. To re-

[1] Graduate School of Engineering, Nagoya University
[2] Institute of Innovation for Future Society, Nagoya University

construct indoor environment with color information, we introduce a Ladybug3 panoramic camera to capture pictures of the surrounding environment simultaneously and then extract RGB information from these images. We use this mobile system to collect data of the real indoor environment for 3D indoor mapping in an underground shopping mall called Unimall near Nagoya station. The registration result verifies practicality, portability and advantage of our mobile system for 3D indoor reconstruction and mapping.

2. Related Work

Many methods for 3D mapping using RGB-D sensors have been proposed. To reconstruct complete scene of an environment, the process of combing many separate point cloud data into one is needed, which is called registration. Iterative Closet Point(ICP), proposed by Besl in [4], is a popular algorithm to solve this problem. ICP repeats calculating the transformation matrix that results the minimum distance of points in overlap parts of two point cloud data. A method called RGBD-ICP, which registers two point clouds using both depth data and RGB data from Kinect is proposed in [3]. KinectFusion implements real-time reconstruction using only depth data from Kinect [7]. For all these methods, it is a precondition that there must be overlap of two point cloud data to register them with ICP algorithm automatically. However, the maximum viewing angle of RGB-D sensors is $70^\circ \times 60^\circ$ with the maximum effective range of 4.5 meters, shown as in Table 1. That is, the area that a frame of RGB-D sensors can cover is less than $10m^2$. Therefore, it will cost a lot of time to collect enough depth data to reconstruct a large indoor environment. And accumulative error caused by multiple ICP registration processes may lead to mis-registration .

LiDAR has been widely used in SLAM [9],[10],[12] and autonomous vehicle [8]. Many applications for SLAM by 2D mapping have been developed [10]. For most of applications for SLAM, 2D LiDAR like Hokuyo UTM-30LX is used. A device called Zebedee is implemented in [9]. Zedebee processes 3D SLAM by mounting a 2D LiDAR and an IMU(Inertial Measurement Unit) on a spring. IMU is essential to recover trajectory for Zebedee . A method for real-time SLAM even without assistance from IMU is proposed in[12] , and a better result with this method can be accomplished if an IMU is available. Velodyne HDL-64E 3D LiDAR sensor, which is the same series as we use in this work, is utilized for SLAM after the trajectory is estimated in [11]. However, purpose of SLAM is generating maps using sparse point cloud for localization rather than dense indoor reconstruction and mapping essentially. Further, all these methods obtain only colourless depth data while attendance of color information could bring better user experience especially for 3D indoor reconstruction and mapping.

A system for indoor mapping with a LiDAR and a panoramic camera, which are same as our system utilizes, is proposed in [13]. However, our system substantially differs from [13]. LiDAR is used for SLAM to reconstructs indoor environment using images from panoramic camera in [13]. This method is practical only for indoor environment that is mainly composed of planes. While our system reconstructs indoor environment mainly using depth data from LiDAR, and images from panoramic are only for complement of color information.

The system we propose combines the advantages of LiDAR sensor and RGB-D sensor. It is capable of measuring distance until 70 meters and generates a colored dense point cloud. A large environment can be reconstructed after registration without any other additional information like GPS or IMU. Additionally it is portable for equipping on a cartographer and moving in indoor environment for data acquisition.

3. Overview of Our System

As shown in Fig.2, our mobile system mainly consists of a Velodyne HDL-32E LiDAR sensor and a Point Grey Research Ladybug3 panoramic camera. Thus, we call it Velobug system (Velodyne + Ladybug). HDL-32E is mounted above Ladybug3, as a consequence, top camera of Ladybug captures nothing. HDL-32E is connected by a ethernet cable and Ladybug3 is connected by a 1394b cable to the laptop in the backpack. Both HDL-32E and Ladybug3 are powered by a 30000mAh battery in the backpack. Earphones, USB switch, bluetooth keyboard and sub-monitor are also connected to the laptop for user interaction.

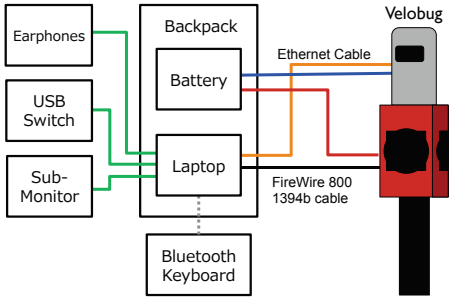


Fig. 2: Structure diagram of our mobile system

Velobug equipped on a cartographer is shown in Fig.3. Aluminum pole is used as the extension for holding Velobug. A customed assistant belt is used to distribute the weight and ensure safety of HDL-32E and Velobug. Earphones connected to the laptop are for notification of operation like “Measurement Started”. Sub-monitor and bluetooth keyboard are for confirming and setting. An optimus mini three keyboard is used as a switch to start/stop acquiring HDL-32E 3D data and Ladybug3 image data.

Detailed components and weight information of Velobug are listed in Table 3. Total weight of Velobug is only about 15kg, which is reasonable for mobile moving. This is event lighter than Google street view trekker backpack which is 42 pounds(about 19 kg). We use aluminum for pole to ensure safety of expensive HDL-32E and Ladybug3, although aluminum pole is very heavy. It is possible to make Velobug lighter by using a better material like carbon fiber instead

Fig. 3: Photo of Velobug system actually equipped on a cartographer

of aluminum for pole. Also a lighter laptop with better performance is also selectable.

As for battery life, the laptop can work for about 3 hours under our measurement conditions. And it can work for another 3 hours with a replacement of 6-cell battery (313 grams). Power consumption of HDL-32E is 12W [14] and Ladybug3 is 7.2W [15] under working conditions of Velobug. A capacity of a mobile battery we used is $30000mAh \times 3.7V \times 50\% = 55.5Wh$, even assuming the efficiency is 50% for 12V output. It is capable to supply power for both HDL-32E and Ladybug3 for $55.5Wh \div (12W + 7.2W) \approx 2.9h$ per mobile battery, which means 8.7 hours by 3 mobile batteries.

In summary, our mobile system is portable (15kg) and posses battery life for at least 6 hours which is long enough for cartography. Also there is still room for improvement in making Velobug lighter.

Table 3: Main components list and weight details of Velobug

Component	Model	Weight(g)
LiDAR	Velodyne HDL-32E	1,300
Panoramic Camera	Point Grey Ladybug 3	2,414
Laptop	HP Probook 6570b	2,600
Mobile Battery×3	JX-R02 30000mAh ×3	470 ×3
Sub-Monitor	CENTURY LCD-8000DA	450
Bluetooth Keyboard	BUFFALO BSKBB13BK	200
Switch Keyboard	Optimus Mini Three	110
Backpack	—	1,200
Assistant Belt	Custom	1,400
Aluminum Pole	Custom	3,300
Other Metal Parts	Custom	800
TOTAL WEIGHT:	—	15,184

4. Increasing Points Density

Velodyne HDL-32E LiDAR sensor measures distances with 32 vertical aligned lasers by Time Of Flight(TOF) method. It provides a 41.34°(+10.67°∼ -30.67°) vertical and 360° horizontal field of view by rotating the head which emits 32 lasers around its vertical axis. For every 360° spin (0.1 second), a frame of point cloud consisting of about 70,000 3D points is generated. Vertical angular resolution can be calculated as $41.3431° \div 31 \approx 1.33°$ and horizontal angular resolution can be calculated as $360° \div (70,000 \div 32 - 1) \approx 0.165°$. Fig. 4 shows an example of point cloud generated by Velodyne. Either from angular resolution or Fig. 4, we could find that points density decreases dramatically as distance increases especially in vertical direction.

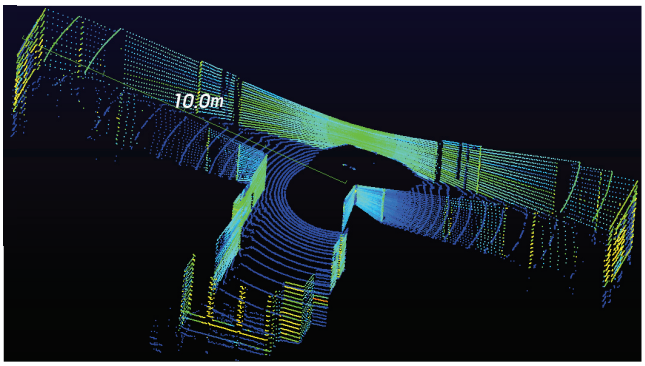
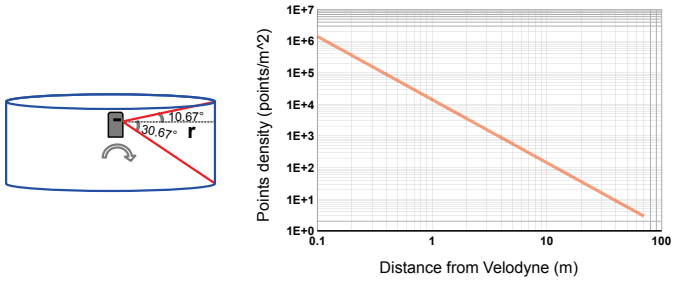


Fig. 4: An example point cloud generated by Velodyne HDL-32E LiDAR sensor

To find relationship between points density and distance approximatively, we assume HDL-32E is in the axis of the cylinder as in Fig.5a. The total area (denoted as A) cover by all 32 laser beams equals $A = 2 \cdot \pi \cdot r[r \cdot \tan(10.67°) + r \cdot \tan(30.67°)] \approx 4.91r^2$. As we have known the total points number for every round is 70,000 (denoted as T_p), the density of points ρ_p can be calculated as $\rho_p = T_p \div A \approx \frac{14256}{r^2}$.

We can know points density is proportional to the inverse square of the distance to HDL-32E shown as in Fig. 5b. Thus, it is necessary to increase points density for dense reconstruction.



(a) (b)

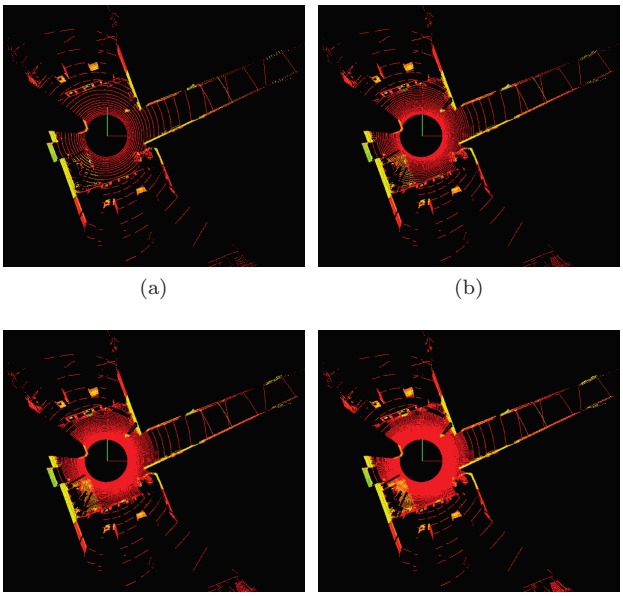
Fig. 5: (a)-Model for estimating points density (b)-Relationship between distance and points density

4.1 Upsampling by Interpolation

We consider interpolating points to increase the points density generated by HDL-32E. As laser_id information is also stored, it is easy to find a pair of two adjacent points from two adjacent laser scanned lines. We could make the point cloud denser by interpolating points in arithmetic mean position. To avoid introducing noise points, a pair

of points is interpolated only if distance between them is less than a threshold value d_{max} defined by the user.

Fig.6 shows the results of upsampling by interpolation for an example point cloud. We can see that points near Velobug became denser observably after the interpolation. However, density of points far from Velobug changed nothing.



(a) (b) (c) (d)

Fig. 6: Upsampling by interpolation, (a)-Raw point cloud, (b)-One point is interpolated, (c)-Two points are interpolated, (d)-Three points are interpolated, $d_{max} = 0.5m$ for (b)(c)(d)

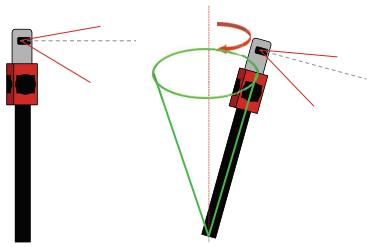
4.2 Upsampling Physically

To make the density of points denser even for those far from Velobug, we think of upsampling physically by changing the way how we collect the data. Usually, we just hold Velodbug to gather data as Fig.7a. For the place we want a dense point cloud like crossway or entrance, we can incline Z-axis of Velobug at an angle and rotate it vertically as Fig.7b.

Fig. 8 shows two pictures of point cloud generated by inclined Velobug at different inclinations in the same place. It is intuitive that result of registration with these two point cloud data covers wider area and possess denser points than point cloud generated by only one frame.

5. Coloring the Points

To color point cloud, we mount a Ladbug3 panoramic camera under Veldoyne. Coordinate systems of Velodyne and Ladybug as shown in Fig.9. We can map a 3D point to a pixel on the image captured by Ladybug3 with a Ladybug API. However, the precondition to do this correctly is that coordinate system of HDL-32E must be unified to coordinate system of Ladybug3. That is, calibration between HDL-32E and Ladybug3 is necessary.



(a) (b)

Fig. 7: (a)-The usual way we use Velobug (b)-Inclined Velobug for denser points

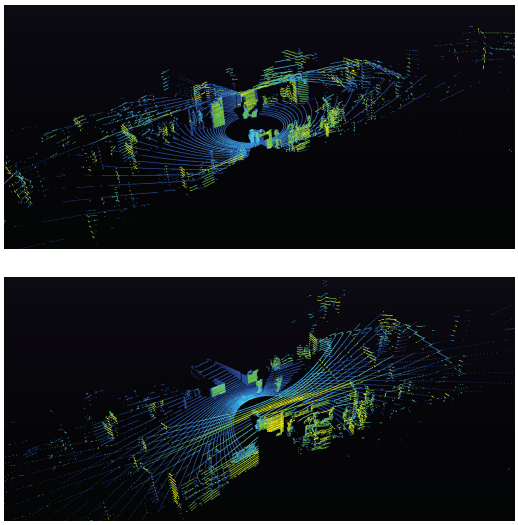


Fig. 8: Example point cloud generated by inclined Velodbug at the same place

We declare notations as follows in this paper:

- $\mathfrak{C}_H$: Coordinate system of HDL-32E with axes $\{x^H, y^H, z^H\}$
- $\mathfrak{C}_L$: Coordinate system of Ladybug3 with axes $\{x^L, y^L, z^L\}$
- M_H: Point cloud of the marker scanned by HDL-32E
- M_L: Image of the maker capture by Ladybug3
- $\boldsymbol{p}_i^{\mathrm{H}} (i = 0, 1, 2, 3)$: 3D coordinates of M_H's vertexes in $\mathfrak{C}_H$, $\boldsymbol{p}_i^{\mathrm{H}} \in \mathbb{R}^3$
- $\boldsymbol{v}_i^{\mathrm{H}} (i = 0, 1, 2, 3)$: 2D coordinates of M_H's vertexes in $\mathfrak{C}_L$ converted from $\mathfrak{C}_H$ by Ladybug API, $\boldsymbol{v}_i^{\mathrm{H}} \in \mathbb{N}^2$
- $\boldsymbol{v}_i^{\mathrm{L}} (i = 0, 1, 2, 3)$: 2D coordinates of M_L's vertexes in $\mathfrak{C}_L$, $\boldsymbol{v}_i^{\mathrm{L}} \in \mathbb{N}^2$

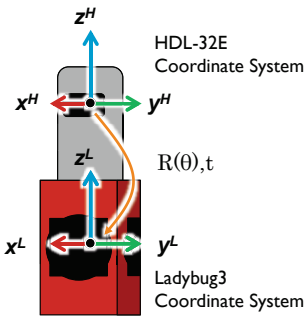


Fig. 9: Coordinate systems of Velodyne and Ladybug3

5.1 Calibration

Calibration of two coordinate systems means calculating the transformation matrix from $\mathfrak{C}_H$ to $\mathfrak{C}_L$. Transformation matrix consists of rotation matrix and translation vector. As z^H is designed in the same line with z^L, it is reasonable to consider that z-axes of two coordinate systems have been aligned even there exists a little error. We only need to calculate rotation angle θ for rotation matrix $R(\theta)$ and translation vector t for x-axis and y-axis.

We use a square paperboard as a marker. We put it near Velobug and start data acquisition. Then the marker will be captured by both HDL-32E and Ladybug3. The original images are shown in Fig.11a and Fig.12a. Then we calculate coordinates of $\boldsymbol{p}_i^{\mathrm{H}}$ and $\boldsymbol{v}_i^{\mathrm{L}}$ respectively. Results of each step for calculating vertexes are shown in Fig.11 and Fig.12.

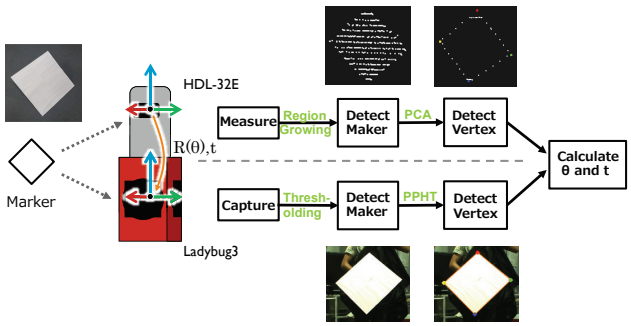
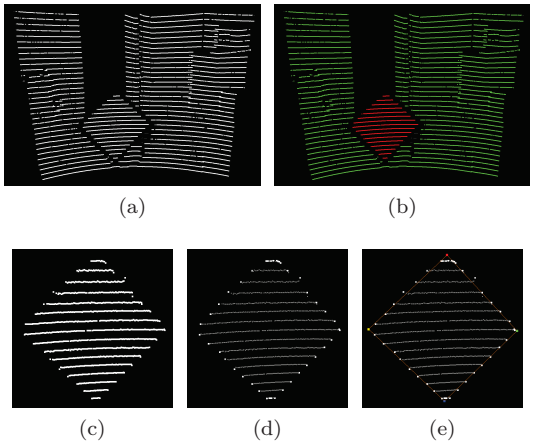


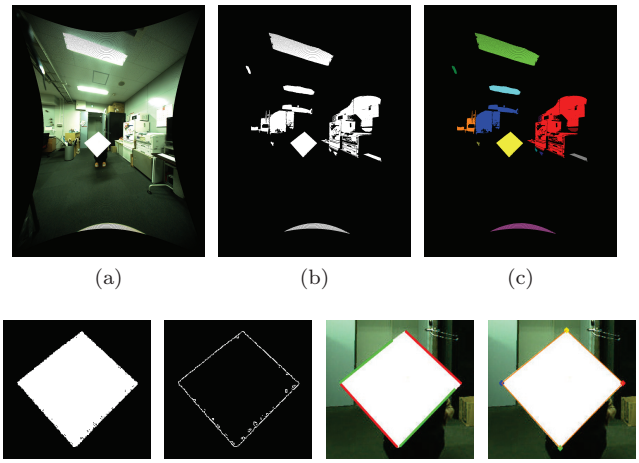
Fig. 10: Flowchart of calculating calibration matrix



(a) (b) (c) (d) (e)

Fig. 11: Marker vertexes detection in point cloud generated by HDL-32E: (a)-Point cloud including marker scanned by HDL-32E, (b)-Clustering result using Region Growing algorithm[16], (c)-Extracted point cloud of marker, (d)-Edge detection by calculating concave hull using Quickhull algorithm[17], (e)- Straight lines are detected using RANSAC[18] and vertexes are calculated

After knowing the value of vertexes, rotation angle can be calculated by Algorithm 1. Firstly, rotation angle θ is initialized as 0. Then, We rotate $\boldsymbol{p}_i^{\mathrm{H}}$ in 3D around z^H by an user defined value $\Delta\theta$. The rotated vertexes are denoted as $\boldsymbol{p}_i'^{\mathrm{H}}$ in $\mathfrak{C}_H$. Then $\boldsymbol{p}_i'^{\mathrm{H}}$ are mapped on the image as $\boldsymbol{v}_i^{\mathrm{H}}$.



(a) (b) (c) (d) (e) (f) (g)

Fig. 12: Marker vertexes detection in rectified image taken by Ladybug3, (a)-the rectified photo including marker from a camera of Ladybug3, (b)Binary image by thresholding for segmentation, (c)-Labeling semgmentation by different color, (d)-Extracted image of marker, (e)-Edge detection using Canny method [19], (f)-Straight lines detection using PPHT algorithm [20], (g)-Calculated vertexes

Euclidean distance e between $\boldsymbol{v}_i^{\mathrm{H}}$ and $\boldsymbol{v}_i^{\mathrm{L}}$ is calculated (Line 11-14). The previous e which is denoted e_{prev} is stored to define whether θ has exceeded the real rotation angle. If $e > e_{\mathrm{prev}}$ which means $\boldsymbol{p}_i^{\mathrm{H}}$ has been over-rotated, $\Delta\theta$ is set to $-0.5\Delta\theta$ (Line 15-17). The while loop is broken when $|\Delta\theta|$ is smaller than $\Delta\theta_{\mathrm{th}}$. We set $\Delta\theta_{\mathrm{init}} = 1^\circ, \theta_{\mathrm{th}} = \frac{1}{256}^\circ$ in this work.

As for translation t, it can also be calculated with the similar method for calculating rotation angle (Algorithm2). We set $\Delta d_{\mathrm{init}} = 0.01m, d_{\mathrm{th}} = 10^{-5}m$ in this work.

It is obvious that the maximum error is $\Delta\theta_{\mathrm{th}}/2$ for rotation angle and $\Delta d_{\mathrm{th}}/2$ for translation vector.

Algorithm 1 Algorithm for calculating roation angle θ

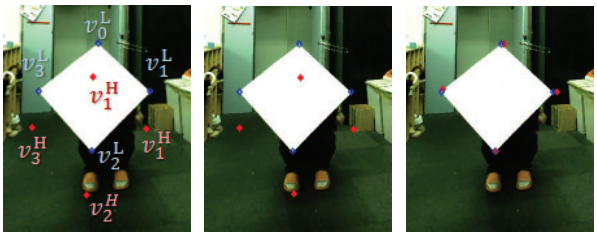
Require: $\boldsymbol{v}_0^{\mathrm{L}}, \boldsymbol{v}_1^{\mathrm{L}}, \boldsymbol{v}_2^{\mathrm{L}}, \boldsymbol{v}_3^{\mathrm{L}}, \boldsymbol{p}_0^{\mathrm{H}}, \boldsymbol{p}_1^{\mathrm{H}}, \boldsymbol{p}_2^{\mathrm{H}}, \boldsymbol{p}_3^{\mathrm{H}}, \Delta\theta_{\mathrm{init}}, \Delta\theta_{\mathrm{th}}$

1: $\Delta\theta \leftarrow \Delta\theta_{\mathrm{init}}$
2: $\theta \leftarrow 0$
3: $e_{\mathrm{prev}} \leftarrow 0$
4: **while** $|\Delta\theta| > \Delta\theta_{\mathrm{th}}$ **do**
5: $\quad\theta \leftarrow \theta + \Delta\theta$
6: $\quad$**for all** $\boldsymbol{p}_i^{\mathrm{H}}(i = 0, 1, 2, 3)$ **do**
7: $\quad\quad$rotate $\boldsymbol{p}_i^{\mathrm{H}}$ to $\boldsymbol{p}_i'^{\mathrm{H}}$ by θ around Z axis
8: $\quad\quad$convert 3D point $\boldsymbol{p}_i'^{\mathrm{H}}$ to 2D point $\boldsymbol{v}_i^{\mathrm{H}}$ by Ladybug API
9: $\quad$**end for**
10: $\quad e_{\mathrm{prev}} \leftarrow e$
11: $\quad e \leftarrow 0$
12: $\quad$**for all** $\boldsymbol{v}_i^{\mathrm{H}}(i = 0, 1, 2, 3)$ **do**
13: $\quad\quad e \leftarrow e + \|\boldsymbol{v}_i^{\mathrm{L}} - \boldsymbol{v}_i^{\mathrm{H}}\|$
14: $\quad$**end for**
15: $\quad$**if** $e > e_{\mathrm{prev}}$ **then**
16: $\quad\quad\Delta\theta \leftarrow -0.5\Delta\theta$
17: $\quad$**end if**
18: **end while**
19: **return** θ

Algorithm 2 Algorithm for calculating translation vector

Require: $\boldsymbol{v}_0^{\mathrm{L}}, \boldsymbol{v}_1^{\mathrm{L}}, \boldsymbol{v}_2^{\mathrm{L}}, \boldsymbol{v}_3^{\mathrm{L}}, \boldsymbol{p}_0^{\mathrm{H}}, \boldsymbol{p}_1^{\mathrm{H}}, \boldsymbol{p}_2^{\mathrm{H}}, \boldsymbol{p}_3^{\mathrm{H}}, \theta, \Delta d_{\mathrm{init}}, \Delta d_{\mathrm{th}}$

for all $\boldsymbol{p}_i^{\mathrm{H}} (i = 0, 1, 2, 3)$ **do**
  rotate $\boldsymbol{p}_i^{\mathrm{H}}$ to $\boldsymbol{p}_i'^{\mathrm{H}}$ by θ around Z axis
end for
$\Delta d \leftarrow \Delta d_{\mathrm{init}}$
$d \leftarrow 0$
$e_{\mathrm{prev}} \leftarrow 0$
while $|\Delta d| > \Delta d_{\mathrm{th}}$ **do**
  $d \leftarrow d + \Delta d$
  for all $\boldsymbol{p}_i'^{\mathrm{H}} (i = 0, 1, 2, 3)$ **do**
    translate $\boldsymbol{p}_i'^{\mathrm{H}}$ to $\boldsymbol{p}_i''^{\mathrm{H}}$ by d on Z axis
    convert 3D point $\boldsymbol{p}_i''^{\mathrm{H}}$ to 2D point $\boldsymbol{v}_i^{\mathrm{H}}$ by Ladybug API
  end for
  $e_{\mathrm{prev}} \leftarrow e$
  $e \leftarrow 0$
  for all $\boldsymbol{v}_i^{\mathrm{H}} (i = 0, 1, 2, 3)$ **do**
    $e \leftarrow e + \|\boldsymbol{v}_i^{\mathrm{L}} - \boldsymbol{v}_i^{\mathrm{H}}\|$
  end for
  if $e > e_{\mathrm{prev}}$ **then**
    $\Delta d \leftarrow -0.5 \Delta d$
  end if
end while
return d



(a) (b) (c)

Fig. 13: Calclulation of roation angle and translation vector, (a)-Blue points are vertexes of marker in Ladybug3, vertexes of marker in HDL-32E are converted on the image as red points (b)-Red points are rotated with rotation angle θ calculated by Algorithm1 (c)-Red points are translated with translation vector t calculated by Algorithm2

5.2 Corresponding

HDL-32E records data separately from Ladybug3. HDL-32E sends data by UDP packet and it is saved as .pcap file in the laptop, while data from Ladybug3 is saved as .pgr file by Ladybug API. Thus, for every frame from HDL-32E we need to find the corresponding image that captured the same scene.

For HDL-32E, elapsed time [μs] since HDL-32E started is included in UDP packet. Also UNIX timestamp of the laptop running Ladybug API is recorded in the .pgr file. We can find the corresponding image for a frame of HDL-32E by finding the image has the nearest elapsed time to that frame (Fig.14).

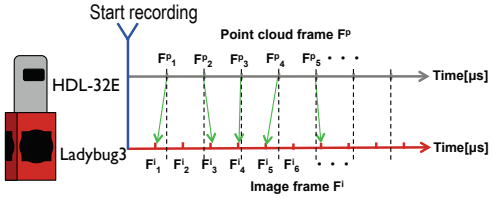
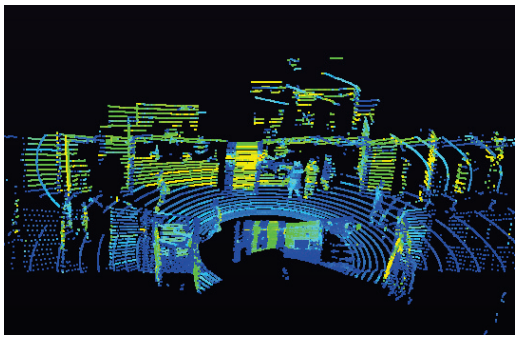
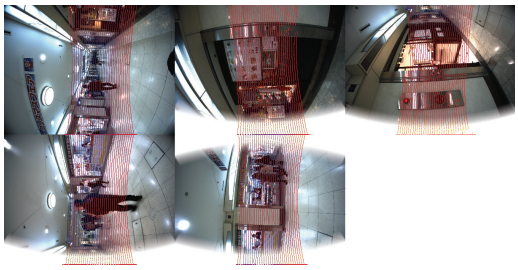


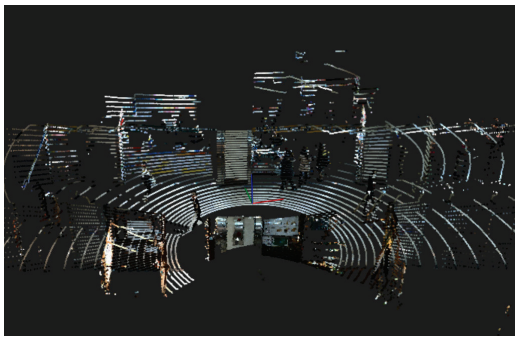
Fig. 14: Correspondence between point cloud frame and image frame



(a)



(b)



(c)

Fig. 15: Color the point cloud (a)-Raw point cloud without RGB information (b)-3D points are mapped to pixels on the corresponding images captured by each camera of Ladybug3, five images are captured by five side cameras separately (c)-Colored point cloud

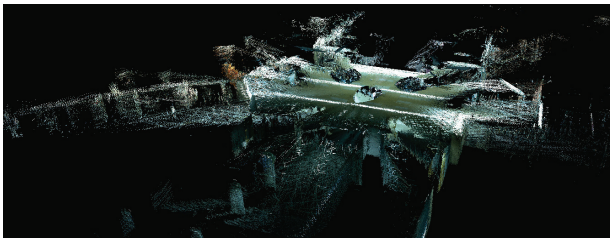
6. Registration Result

Although one frame of the data acquired by Velobug has covered very wide area, density of the points those are far from Velobug is very sparse. As mentioned above, to combine a dense point cloud, it is necessary to register with other frames acquired at different inclination as Fig.7b. Since Generalized-ICP is thought to perform better in robustness

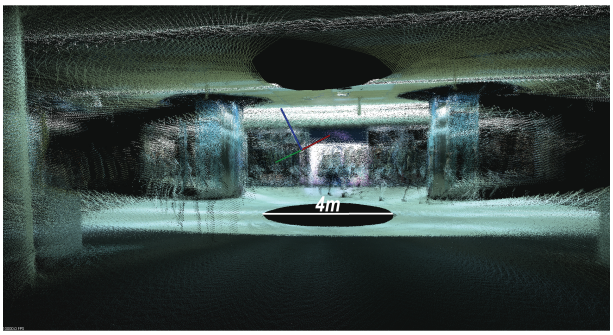
than the standard ICP. In this work, we use Generalized-ICP [5] for registration.

A example result of registration is shown in Fig.16. This point cloud is generated from 37 frames with Generalized-ICP method after upsampling for each frame. From Fig.16 we can see it is well registered for all 37 frames. And points became much denser than that from only one frame especially in intervals between adjacent laser beams. The black circle in the center is the blind area of Velobug. This blind area can also be covered as cartographer moves.

It takes only about 1 minute to acquire enough data that generates point cloud of the environment with a similar scale as shown in Fig.16. It is obvious that Velobug works more effectively at indoor mapping.



(a) Registered point cloud (outside)



(b) Registered point cloud (inside)

Fig. 16: Registration result with Generalized-ICP

7. Conclusion and Future Work

In this paper, we proposed a mobile system for 3D indoor mapping with Velobug HDL-32 LiDAR sensor and Ladybug3 panoramic camera. Benefited from long effective range of LiDAR, this system is particularly suitable for indoor mapping in large environment. Although sparsity becomes very severe as measurement distance increases, this can be solved by upsampling with interpolation and upsampling physically. Furthermore, with the help of Ladybug3, color information can also be captured. The experimental result of an underground shopping mall verified that our system is portable and effective for data acquisition in indoor environment, and dense point cloud with color information could be generated by this system.

However, traces of pedestrians filling in the inside of the point cloud can be seen in Fig.16b. These traces are noisy and need to be cleared for quality mapping. Registration for multiple frames acquired at different inclination makes points denser, however denser points cause more computation for mapping. Thus, our future work will focus on detection and elimination of pedestrians in point cloud and polygonalization of point cloud .

References

[1] http://indoorgml.net (accessed 2015.03.31)
[2] Joon-Seok Kim, Sung-Jae Yoo, and Ki-Joune Li,, "Integrating IndoorGML and CityGML for Indoor Space," *Web and Wireless Geographical Information Systems*, pp.184-196, 2014.
[3] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *The International Journal of Robotics Research*, Vol.31, pp.647-663, 2012.
[4] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, pp.239-256, 2012.
[5] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," *Robotics: Science and Systems*, 2009.
[6] P. Biber, H. Andreasson, T. Duckett, and A. Schilling, "3D Modeling of Indoor Environments by a Mobile Robot with a Laser Scanner and Panoramic Camera" *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004
[7] IS. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. A. Newcombe, P. Kohli, S. Shot- ton, J. Hodges, D. Freeman, A. J. Davison, and A. Fitzgibbon, "Kinect Fusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp.559, 2011.
[8] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J.Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: Systems and algorithms," *IEEE Intelligent Vehicles Symposium, Proceedings*, pp.163-168, 2011.
[9] Michael Bosse, Robert Zlot, P. Flick, "Zebedee: Design of a Spring-Mounted 3-D Range Sensor with Application to Mobile Mapping," *Robotics, IEEE Transactions on*, vol.28, no.5, pp.1104-1119, Oct. 2012
[10] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on* , pp.155-160, Nov. 2011
[11] F. Moosmann, and C. Stiller, "Velodyne SLAM," *Image, Radar, Lidar Signal Processing,Vehicle Environment Perception* , pp.393-398, 2011
[12] J. Zhang, and S. Singh, "LOAM : Lidar Odometry and Mapping in Real-time," *Robotics: Science and Systems* , 2014
[13] P. Biber, H. Andreasson, T. Duckett, and A. Schilling, "3D Modeling of Indoor Environments by a Mobile Robot with a Laser Scanner and Panoramic Camera," *IEEE/RSJ International Conference on Intelligent Robots and Systems* , 2004
[14] Veldoyne LiDAR, Inc. "USER'S MANUAL AND PROGRAMMING GUIDE, HDL-32E," 2012 Getting Started LADYBUG3 1394b Spherical Vision System
[15] Point Grey Research, Inc. "Ladybug3 Getting Started Manual," 2011
[16] T. Rabbani, F. A. van den Heuvel, and G. Vosselmannv, "Segmenting point clouds by region growing using normals and curvature estimation," *IEVM06* , 2006
[17] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa, "The Quickhull algorithm for convex hulls," *ACM TRANSACTIONS ON MATHEMATICAL SOFTWARE* , vol.22, no.4, pp.469-483, 1996
[18] Martin A. Fischler and Robert C. Bolle, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM* , vol.24, no.6, pp.381-395, 1981
[19] John Canny. "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* , vol.PAMI-8, no.6, pp.679-698, 1986
[20] J. Matas, C. Galambos, and J. Kittler, "Robust Detection of Lines Using the Progressive Probabilistic Hough Transform," *Comput. Vis. Image Underst.* , vol.78, no.1, pp.119-137, 2000
[21] W. Grant, R. Voorhies, and L. Itti, "Finding planes in LiDAR point clouds for real-time registration," *IEEE International Conference on Intelligent Robots and Systems* , pp.4347-4354, 2013