FUSION OF LIDAR 3D POINTS CLOUD WITH 2D DIGITAL CAMERA IMAGE

by

JUAN LI

A thesis submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

2015

Oakland University
Rochester, Michigan

APPROVED BY:

Jia Li, Ph.D., Chair	Date
Manohar Das, Ph.D.	Date
Andrew Rusek, Ph.D.	Date

ACKNOWLEDGMENTS

It has taken one year for me to complete my research for my master's thesis. Most importantly, I would like to thank my advisor, Dr. Jia Li, for introducing me to the world of digital image processing, and for her guidance, patience, and support throughout my graduate studies. I really appreciate her weekly meetings to discuss my questions about my research. She met with me every week, even when she went to Ohio to do research in the summer. The meetings were very useful. She guided me toward the right path when I was stuck on a question and she also gave me hints for the next step. I want to thank for her patience when explaining the most trivial and basic questions and for providing me with an expensive sensor and other equipment to complete my research.

I also would like to thank Dr. Manohar Das and Dr. Andrew Rusek for being a part of my graduate committee. Professor Das, as my advisor, gave me guidance in the electrical engineering field. Professor Rusek helped me during the course of my education at Oakland University. I would like to thank Xiang He and Kapil in my research group for experimenting with me. Finally, I would like to thank my family who supported me unconditionally to go abroad to seek a higher education.

Juan Li

ABSTRACT

FUSION OF LIDAR 3D POINTS CLOUD AND 2D DIGITAL CAMERA IMAGE

by

Juan Li

Adviser: Jia Li, Ph.D.

Three-dimension imagery is popular these days. There are many methods for fusing multi-sensors. This thesis develops a rigid-body extrinsic calibration and registration for multi-modal sensor data fusion for 3D mapping. The time-aligned data, 3D points cloud and their intensity information from LiDAR, and texture and color from the camera, are generated by scanning the same physical scene in different manners. The range sensor and camera capture the features of fiducial targets to generate a transformation matrix. Then, two types of images, 2D image from a camera and a 3D points interpolated to a 2D intensity image, can be aligned. Registration of color information from panoramas to 3D points clouds from the LiDAR range sensor are needed to consider the correspondence between pixel coordinates of the intensity image matched with panoramas and spatial coordinates of LiDAR points. Aligning the adjacent locations one-by-one, we can generate a 3D map of a hallway in a large building. This thesis solves some of the common problems such as extrinsic calibration, scan registration, and 3D points alignment. Although this research is dedicated to indoor mapping, it can be generalized to outdoor mapping by changing to a large range LiDAR sensor.

TABLE OF CONTENTS

TABLE OF CONTENTS-CONTINUED

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES-CONTINUED

CHAPTER 1

INTRODUCTION

Computer vision-based approaches are popular when applying to combine various sensors in one system. The method of computer vision can be used for a range sensor multi-sensor system. Combining LiDAR with a camera using the computer vision method modeling is a relatively new technology. During the past decades, LiDAR emerged as a new technology for quickly capturing data of distance and reflectivity. The data of LiDAR can provide more details by fusing with camera images. Fusion of LiDAR and camera data has many applications, such as virtual reality, autonomous driving, and machine vision. The applications also include fusion of optical imagery, using LiDAR to construct a 3D model with color, texture information, and so on. For example, one can track his/her location when walking in a museum. One can know what a building’s interior looks like before visiting it. One way for car operation without drivers is to use LiDAR to detect distance and a camera to guide it.

There are two common types of LiDAR sensors for scanning the environment: 2D LiDAR sensors like UTM-30LX-EW and 3D LiDAR sensors like Velodyne. Recent work uses 3D LiDAR to scan the environment, because it can directly generate a 3D points cloud. The omnidirectional camera is widely used for providing a wide view image [8, 10]. In this work, for the fusion of a LiDAR sensor and a camera, two types of sensors are installed on a common platform to provide complete information. Recently, high quality 2D LiDAR is used in combination with a camera for a moving machine to detect objects and reconstruct them. Three important modules mounted on a mobile platform are: (1) range sensor (2D LiDAR), (2) moving machine (servo), and (3) optical camera. We get two types of data from a multi-sensor system. The 2D LiDAR sensor can only scan a horizontal azimuth. Although using a 3D LiDAR would be convenient for fusing the data, a 2D LiDAR have many advantages such as light weight, low cost, and so on[6, 10]. We make the LiDAR sensor cover a portion of the vertical field of view by mounting the 2D LiDAR on the top of the servo. The servo can hold the 2D LiDAR sensor and stop at different elevation angles. Once the servo completes its 180^o rotation, the connected PC will collect a list of data from the 2D LiDAR in different elevation angles. Then, a list of data is converted from spherical coordinates to cartesian coordinates and 3D points are generated for the front of view. The digital camera has a limitation that only has a narrow field of view for matching. The camera images need to be put into the Panorama Maker 6 software to generate the panorama. There are three panoramas related to the positions of left, middle, and right. Then, the three panoramas can cover the most field of view in front of the LiDAR-camera system.

Next, extrinsically calibrating the LiDAR with the camera is the key for fusing multi-sensor data. Extrinsic calibration is the process of estimating the rigid-body transformation between the two sensors' coordinate systems. Substantial prior work has been done on extrinsic calibration of a multi-layer LiDAR and a camera is useful in outdoor environments [29]. Once we obtain the data from these two sensors independently, this captured complementary information needs to be extrinsically calibrated. The contribution of transformation matrix from Hartley, Richard, and Andrew

Zisserman can be used in the research [18]. A variety of methods have been developed to address the LiDAR-camera extrinsic calibration problem. A fiducial target for extrinsic calibration is necessary. Among them, Xiaojing Guo and Ying Lin et al. use a special calibration rig such as a trihedral calibration rig [1, 11]. Alismail et al. completes extrinsic calibration by making use of a circle [21, 29]. Park and Yoonsu et al. rely on a board pattern [3, 26]. Lipu Zhou and Florez et al. use a planar checkerboard pattern to calibrate multi-sensors [12, 22, 24, 27]. Corners of the pattern are first detected in images and used to determine the poses of planes in camera frames. Meanwhile, 3D points falling on the checkerboard are taken into consideration to estimate the poses of the fiducial target in LiDAR frames. In multi-sensor fusion, two different sensors are combined to estimate the surface features of the objects. To combine two types of data from two sensors, we need to find the correspondence between these two data, which means the relationship between the 3D points cloud and 2D image. Using the geometric constraint of the checkerboard pattern in several LiDAR-camera observations, the extrinsic calibration problem is formulated as the RANSAC problem.

In this work, because the LiDAR is rigidly connected to a camera, the transformation is rigid while the relative positions of the camera and LiDAR are settled. It is difficult to find the transformation matrix that allows the 3D points from the range sensor to be projected directly to a 2D camera image. The easy way for matching the LiDAR data with the camera image is to project 3D points from the range sensor into the 2D points and match the two 2D images. After 3D points of intensity information obtained from the LiDAR scanner are projected into 2D points, the 2D points with intensity value are interpolated into an intensity image in grayscale. We detect the same fiducial target in both intensity image and panorama. The correspondence between the extracted features from the LiDAR intensity image and panorama is established by the RANSAC method. Budge and Scott et al. also use the RANSAC algorithm to complete fusion of different types of multi-sensors [25, 32]. Using the RANSAC algorithm to calculate the transformation matrix means mapping the greatest number of point of pairs between the panorama and LiDAR intensity image based on a distance threshold. This method repeats generating solutions estimated from a minimal set of correspondences. The correspondence for the image point pairs that is generated from these two sensors is the projective relation. A common camera is used in the research instead of using an omnidirectional camera [8, 10]. For obtaining more points with color and texture information, three panoramas and related three transformation matrices are needed for the research. We get all the transformation matrices for fusing panoramas with the LiDAR intensity images by putting fiducial targets in three positions. We stitching these three panoramas together to get a wide view for matching the LiDAR intensity image with the panorama. The rigid transformation matrix can be used for the rest of the life of the match LiDAR and camera data in Dodge Hall.

We complete the registration of panorama texture information to LiDAR 3D points. The texture information is useful to complete registration of the textured 3D points cloud [13, 19]. There has been a large amount of research about registration of the multi-view optical images with LiDAR imagery and other geometric models. Liu et al. applied structure from motion to a collection of photographs to infer a sparse set of 3D points, and then performed 3D-3D registration [31]. The 2D-3D registration also has a wide application [23, 29]. In this work, once we get the transformation matrix, the 2D-2D correspondence between the two sets of points is known. According to the correspondence between image pixel indices and image spatial coordinates, the 2D projected points can get the color information from the textured LiDAR intensity image. The nearest integer coordinate of the 2D projected point is used to get the color value from the transformed panorama. The textured 2D points can be projected back to 3D points by the sequence of the points list. Then, the 3D points with realistic texture and color information are generated for each detecting location.

There are many applications for fusion of the data of a multi-sensor system. This technology can be used in robotics, vehicles, and so on to generate maps of the environment. Pandey and McBride et al. equip the devices on a Ford car to map the outdoor environment [9]. Arefi and Engels et al. scan the data for a large building [4, 20, 28, 31]. In this work, we generate the hallway of Dodge Hall into a 3D map. We mark measuring locations before scanning for preparation. We set a certain distance for the measuring locations and measured the size of the hallway. After getting the 3D points with texture information from panorama, we align two sets of textured 3D points cloud to generate the 3D hallway map. The overlap of two adjacent 3D textured points cloud can help us remove the blank points. The range of each textured 3D points cloud are selected for aligning to a 3D map. We align the textured 3D points cloud from two opposite directions to generate a closed hallway. After generating the textured 3D points cloud of the four combined hallways of Dodge Hall, we put the data with coordinates and color

RGB information into MeshLab. Then we generate a realistic 3D visual map for Dodge Hall.

This thesis is organized as follows: In Chapter 2, we briefly introduce the LiDAR sensor and the digital camera. This chapter also includes the analysis of the data from the LiDAR sensor and the digital camera. Extrinsic Calibration of LiDAR and camera data by using fiducial targets is described in Chapter 3. Chapter 4 covers the registration of textured 3D points cloud and the alignment of measuring locations to generate the map of hallway. The import of data into MeshLab software and the 3D model visualization is also demonstrated in Chapter 4. In Chapter 5, the conclusion and future work shows the work for this research.

CHAPTER 2

RANGE SENSOR AND CAMERA

2.1 LiDAR

2.1.1 LiDAR

LiDAR is the abbreviation of Light Detection and Ranging. It measures the distance by illuminating a target with a laser and analyzing the reflected light. LiDAR can get an accurate result in measuring distance. It can also get intensity values for other uses, such as matching with a camera image. Hokuyo UTM-30LX-EW LiDAR sensor is used in my research, because it is commonly used for indoor environments. This sensor is a type of 2D LiDAR sensor and has many advantages [6]. It is a low-cost range sensor among 2D LiDAR sensors. It provides accurate and dense data of indoor environments. It is also easy to mount because of its small size and light weight. This type of LiDAR is designed for indoor scanning, autonomous navigation, and other applications. The 2D LiDAR can scan a horizontal plane which is vertical to the head of the LiDAR device. The Hokuyo UTM-30LX-EW LiDAR sensor is rated to provide usable returns up to 30 meters. The LiDAR sensor is shown in Figure 2.1.

2.1.2 LiDAR with Servo

The LiDAR's detection range is up to 30m and its wide-angle is up to 270°. This LiDAR sensor is used for indoor detection. We set the wide-angle range to be 180° and there are 721 sampling points for each horizontal plane. Compared to 3D LiDAR, such as Velodyne Laser Scanner, the Hokuyo UTM-30LX-EW 2D LiDAR sensor cannot scan

Figure 2.1: Model of LiDAR (UTM-30LX-EW).
(https://www.hokuyo-aut.jp/02sensor/07scanner/download/products/utm-30lx-ew/)

data in a vertical direction [6]. Compared to the 3D Velodyne LiDAR, it costs less, needs lower power and has less weight. The size of the Hokuyo LiDAR is one fourth of the size of the Velodyne LiDAR sensor. The 2D scanning sensor is mounted on the top of a moving servo named Turret Widow X. The Hokuyo LiDAR is quite light and the Turret Widow X servo allows it to rotate. The moving machine can hold the weight of the LiDAR sensor and its maximum holding weight is 25.5 kg from its specification. The system of moving machine and scan sensor can get the 3D data in front of the detect system. We accomplish the vertical sweep by using a servo. The 2D LiDAR is mounted on the top of a servo. We set the vertical angle range to 180° and using 127 steps to scan approximately 180° in vertical wide view. We want the range sensor scans in vertical direction per 1^{o}. Unfortunately, we can only get 127 steps and it’s approximately one scan per 1.5^{o}. Figure 2.2 shows Turret Widow X servo and LiDAR mounted on it.

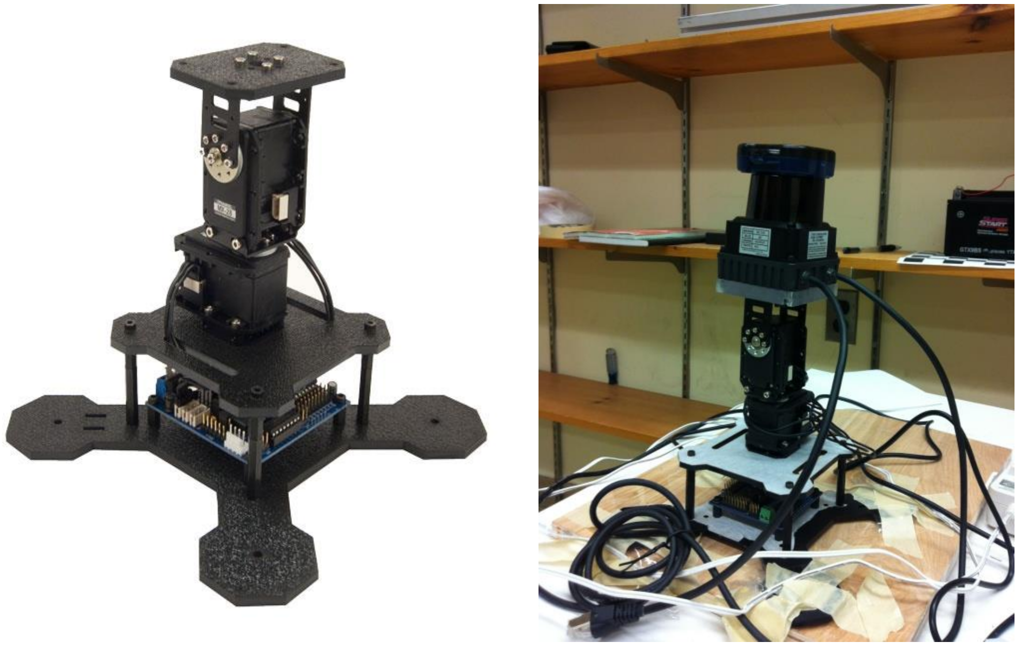


Figure 2.2: Turret Widow X servo (left); LiDAR with Turret Widow X servo (right). (http://www.trossenrobotics.com/p/WidowX-robot-turret.aspx)

Arduino is an open-source computer hardware and interactive device that can sense and control the physical world. Arduino, in my research, is programmed to control the servo. The servo has two moving directions: horizontal and vertical. For a horizontal plane, it can be programmed moving from 90^o to 90^o in a resolution of 0.88^o. The moving steps and directions are programmed according to our requirement. The LiDAR data needs more steps to generate 3D points than the steps of the camera to create the panoramas.

The entire unit can scan as a 3D LiDAR scans the field of the front view precisely. It can provide 180^o horizontal azimuthal field of view by the Hokuyo 2D LiDAR sensor and 180^o vertical azimuthal field of view by the Robot Turret Widow X servo moving the LiDAR in a vertical direction. The LiDAR data is obtained by moving a 2D LiDAR. The 3D LiDAR uses the same method to obtain data. For the Velodyne Laser Scanner, once the head of the LiDAR completes a full horizontal rotation, each scanner needs to vertically sweep a cone in space [30]. The principle of operation for Velodyne 3D LiDAR is shown in Figure 2.3.

2.1.3 <u>3D Points Cloud of LiDAR System</u>

There are many methods for using a line-scan 2D LiDAR sensor to generate the 3D points. It can sweep the scanning plane in a horizontal or vertical direction. It is better to rotate the line-scan LiDAR horizontally. There are two advantages for making the servo move vertically. The first is that the camera will set in three positions in the horizontal plane of the camera. The second advantage is that it will avoid the block from the mounting device for the camera, and then the LiDAR can scan 180^o without obstacle.

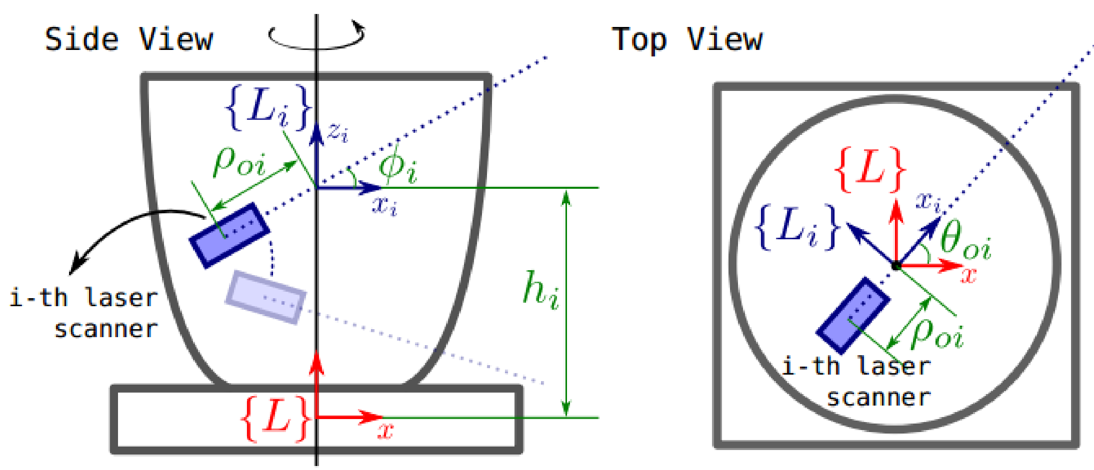


Figure 2.3: Internal structure of the Velodyne 3D LiDAR.
(Mirzaei, Faraz M., Dimitrios G. Kottas, and Stergios I. Roumeliotis. "3D LIDAR–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization." The International Journal of Robotics Research 31.4 (2012): 452-467.)

We use the term ‘scan’ to describe a line-scan LiDAR to get the data from a scanning plane and the term ‘sweep’ to describe the LiDAR rotation with the servo to get the data in a vertical direction. We set the pivot of the LiDAR system (includes LiDAR and servo) to be the center of the servo, because we can scan the 3D data as a spherical model. The distance between the head of the LiDAR sensor and the center of the LiDAR-servo system needs to be measured. The distance measured is 101 mm in this experiment. We use the horizontal field of view of 180^o for Hokuyo LiDAR sensor and the range for the front view is enough for the fusion with the camera color image in the next step. Figure 2.4 is the line-scan obtained by LiDAR in the scanning plane. The software in the host PC makes the data scan by Hokuyo LiDAR transparent. The green region in Figure 2.4 is the distance information obtained from the LiDAR range sensor and the purple region is the intensity information of the objects. The scan management is the LiDAR-servo system and there are three parts the Hokuyo line-scan LiDAR sensor, the Turret Widow

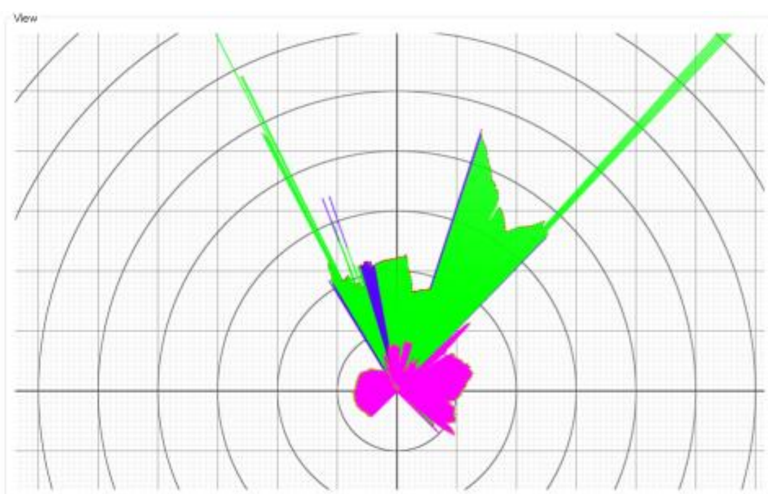


Figure 2.4: LiDAR data display in software.
(https://www.hokuyo-aut.jp/02sensor/07scanner/download/data/UrgBenri.htm)

X servo machine, and a microcontroller board connected to the PC. We need to set the angular velocity of the servo and control the time between the two adjacent positions. Because the Hokuyo LiDAR sensor scans a scanning plane very fast, it only needs a little time between two adjacent positions. We can ignore the servo's rotational speed. It has a delay time at the beginning and the waiting time is 7 seconds. We set 3 seconds as the delay time between two adjacent positions of the servo, and it will get a higher resolution in scanning when the servo rotates with delay. The delay time is even, and then the code for automatically getting a sequence scan data will be easy. The scan system obtains data by scanning over 180^o horizontally with LiDAR and sweeping over 180^o vertically using the servo, as is shown in Figure 2.5. It can capture all the area in front of the sensor.

Converting spherical coordinates to cartesian coordinates means converting the information of distance, azimuth, and elevation to XYZ coordinates. The conversion

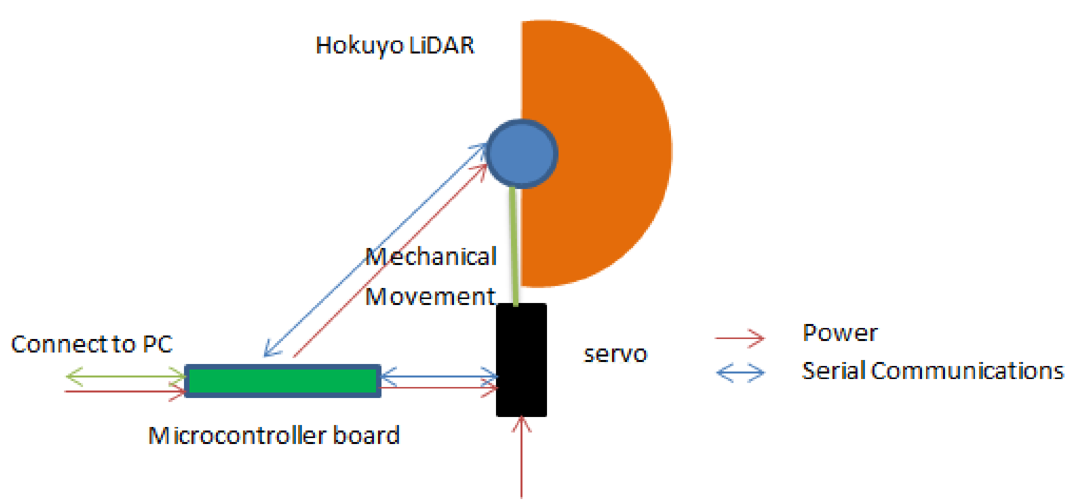


Figure 2.5: Connection for the components of detection system.

generates a 3D points cloud in cartesian coordinates [10].The X coordinate is the height of the scan view, the Z coordinate describe the width of the object to the principle line, and the Y coordinate is the distance from the center of the sensor system to the plane that the object located on. The points are converted from spherical coordinates to cartesian coordinates, as shown in Figure 2.6. The converting equation (2-1),(2-2),(2-3) is shown as follows.

$$x = r * \cos\theta \tag{2-1}$$

$$y = r * \sin\theta * \cos\beta \tag{2-2}$$

$$z = r * \sin\theta * \sin\beta \tag{2-3}$$

where β means the value of azimuth and θ describes the polar angle.

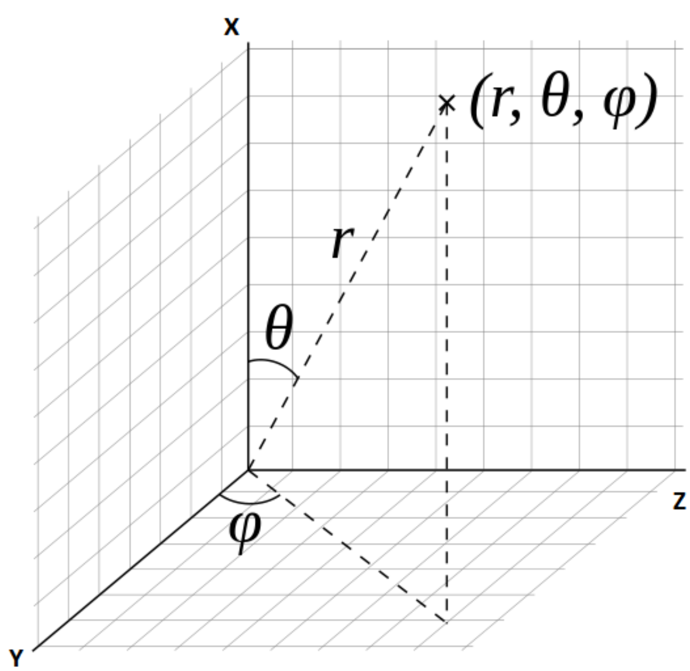


Figure 2.6: Sperical coordinates.
(http://www.ask.com/wiki/Spherical_coordinate_system?o=2800&qsrc=999&ad=double Down&an=apn&ap=ask.com)

2.2 Image Sensor

2.2.1 Nikon S3000 Camera

The Nikon S3000 camera sensor shown in Figure 2.7 (left) is used to get texture information from the environment. The digital camera captures images the size of 640 x 480 in RGB image format (.png file). The size of 640 x 480 pixels is enough for this research. The camera is an ordinary type. It can be used easily, and more cheaply than the Electro-Optical sensor. We mount the camera on the top of the LiDAR so that it can move with the LIDAR at the same time, as shown in Figure 2.7 (right).

2.2.2 Image Panorama

2.2.2.1 Image Distortion

Assuming we know the intrinsic parameters, we get the extrinsic parameters for the fusion of the LiDAR 3D points cloud and the camera image. Using MATLAB

Figure 2.7: Nikon S3000 camera (left); Camera integrated LiDAR (right).

Camera Calibration App, we can find the intrinsic parameters for the camera, radical distortion, and tangential distortion. Applying these parameters to the detected images, we can get the undistorted images.

2.2.2.2 Introduction of Camera Model

The list of internal parameters:

- **Focal length:** The focal length in pixels is stored in the 2x1 vector $\boldsymbol{f_c}$.
- **Principal point:** The principal point coordinates are stored in the 2x1 vector $\boldsymbol{cc}$.
- **Skew coefficient:** The skew coefficient defining the angle between the x and y pixel axes is stored in the scalar **alpha_c**.
- **Distortions:** The image distortion coefficients (radial and tangential distortions) are stored in the 5x1 vector $\boldsymbol{k_c}$.

2.2.2.3 Run Camera Calibration App

After running the Camera Calibration Application shown in Figure 2.8, we can get intrinsic parameters and both radial and tangential distortion coefficients to calibrate images. If some predicted corners are far from the related real grid, the cause is image distortion. The green circles are the detected corners, and the red points are the guess points. The left side is the average error in each image and the position of the checkerboard image relative to the position of the camera. That can verify whether the images from all directions have been captured or not. The size of the checkerboard in my research is 57.5 mm.

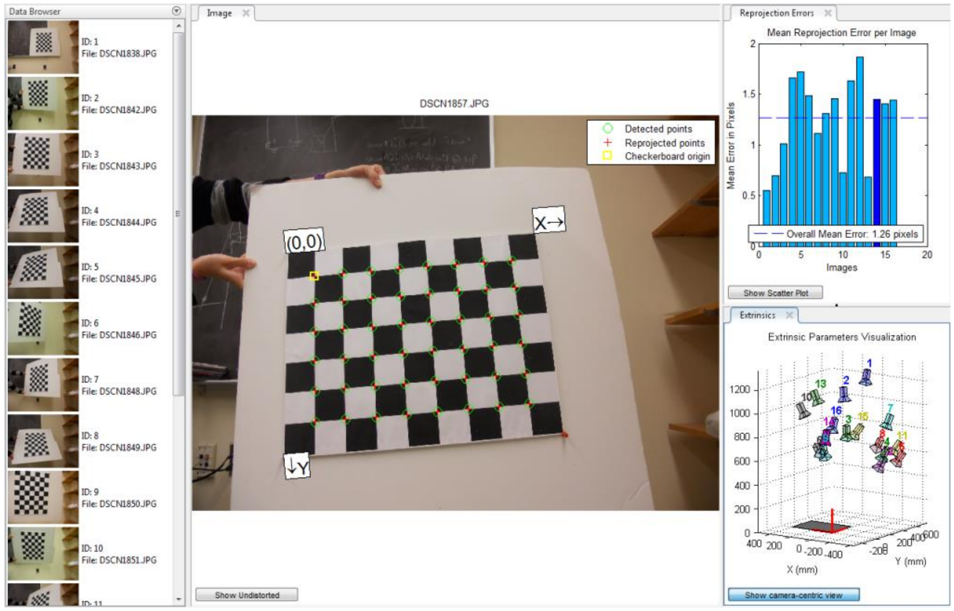


Figure 2.8: Checkerboard points detection in Camera Calibration App.

The MATLAB function $undistortImage$ is used here to get a better image without distortion. We need to delete some large error images to get a better result.

2.2.2.4 Comparing the Original Image and Image without Distortion

After getting distortion coefficients, intrinsic parameters, and extrinsic parameters, we can use them to calibrate images. The calibrated camera images are shown in Figure 2.9 and Figure 2.10.

Comparing these two images of the checkerboard, the edge line becomes straighter after camera calibration. It will obviously show the edge of the checkerboard and the edge of the white board.

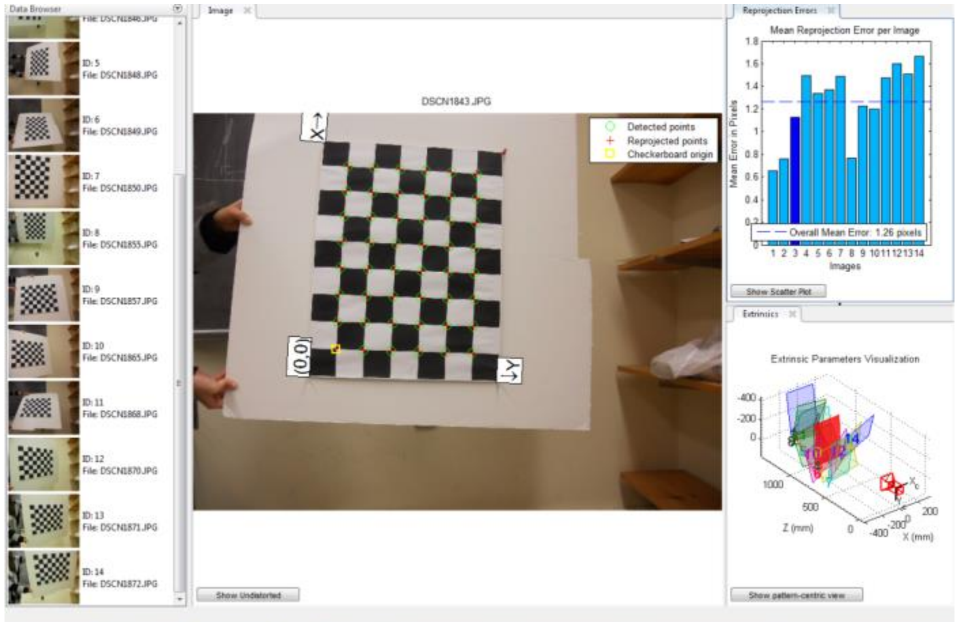


Figure 2.9: The detected corners in checkerboard.

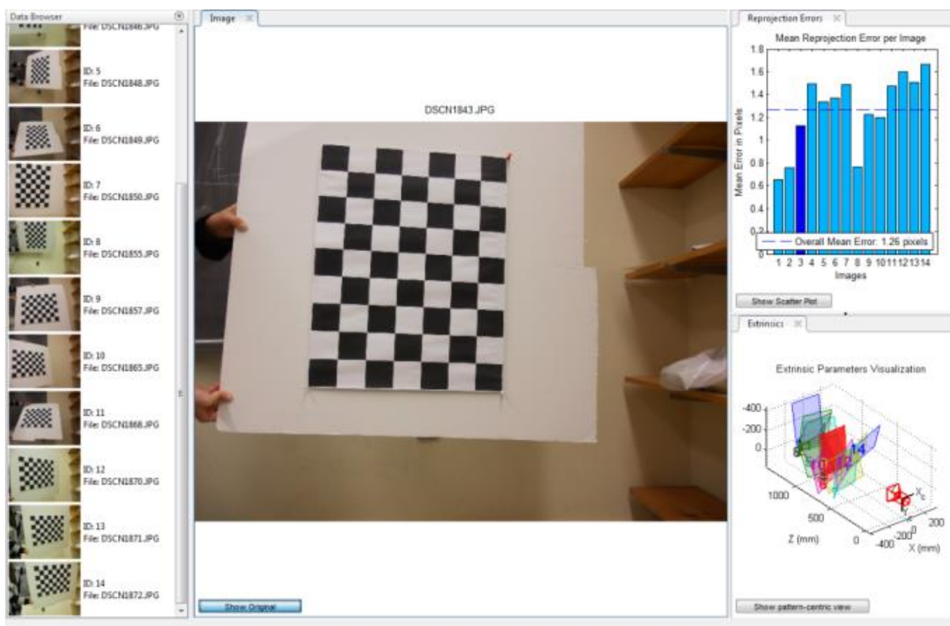


Figure 2.10: The positions of checkerboard for camera.

For the radial factor, we use the following formulas (2-4),(2-5):

$$x_{corrected} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{2-4}$$

$$y_{corrected} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{2-5}$$

where k_n is the n^{th} radial distortion coefficient; r is the radial length in original pixel point coordinate; (x, y) is the point in pixel coordinates; $(x_{corrected}, y_{corrected})$ is the point in corrected image. It is worth noticing that we let $r^2 = x^2 + y^2$, then there isn't odd power for radial.

So for a pixel point at (x, y) coordinates in the input image, its position on the corrected output image will be $(x_{corrected}, y_{corrected})$. The presence of the radial distortion manifests in form of the "barrel" or "fish-eye" effect.

Tangential distortion generated by imaging lenses is not perfectly parallel to the imaging plane. It can be corrected via the formulas (2-6),(2-7):

$$x_{corrected} = x + [2p_1xy + p_2(r^2 + 2x^2)] \tag{2-6}$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2xy] \tag{2-7}$$

where p_n is the n^{th} tangential distortion coefficient.

So we have five distortion parameters which in OpenCV are presented as a row vector with 5 columns:

$$Distortion_{coefficients} = (k_1\ k_2\ p_1\ p_2\ k_3).$$

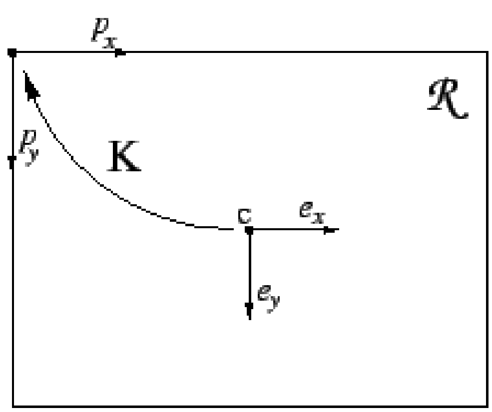
Now for the unit conversion we use the following formula (2-8):

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2-8}$$

The presence of w here is explained by the use of a homograph (and $w = Z$). The unknown parameters are f_x and f_y (camera focal lengths), and (c_x, c_y) are the optical centers are expressed in pixels coordinates. For both coordinates if a common focal length is used with a given aspect ratio (usually 1) a, then $f_y = f_x * a$ and in the upper formula (2-8) we will have a single focal length f.

The matrix containing these four parameters is referred to the camera matrix. While the distortion coefficients are the same no matter what's kind of the camera resolutions we used, these four parameters should be scaled along with the current resolution from the calibrated resolution.

Figure 2.11 shows the camera intrinsic calibration. For most camera models, the size of pixel is almost perfectly rectangular. Furthermore, the principal point c in Figure 2.11 is often located in the center of the image. Such assumption can be used in the Camera Calibration Application, certainly to get a suitable initialization for more complex iterative estimation procedures.



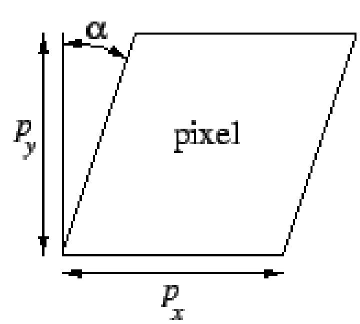


Figure 2.11: Camera intrinsic calibration.

2.2.3 Panorama Generation Software

2.2.3.1 Panorama Maker 6

The Nikon S3000 digital camera has a limited range of width and length for the 3D model. The LiDAR will scan 180^o of view which is much wider than the camera image view. We need to take 13 images first and stitch them into a camera panorama in one direction. The camera panorama will have a wider range for the front view of width and length. Then, we will get the other two directions (left and right) panorama image applying with the same method.

The Panorama Maker 6 software can stitch images in five modes, i.e. auto, horizontal, 360, tile and vertical. In my research, the images captured by camera with servo moving in vertical direction. We capture images per 10^o in vertical direction. The images are pre-calibrated from the MATLAB Camera Calibration App. The intrinsic parameters of Nikon S3000 camera that we used in the research are all known. Finally, the panorama is generated by stitching 13 images in vertical mode.

2.2.3.2 Stitching Images

The last step for generating panorama is customizing the panorama image. The size and the view can be changed by your selection. After select the view of the panorama image, the panorama can be exported to different formats like JPEG, BMP and so on. Figure 2.12 shows the panorama is generated by stitching 13 images.

Figure 2.12: Generation of the panorama.

CHAPTER 3

EXTRINSIC CALIBRATION OF LIDAR AND CAMERA

3.1 Introduction

There are many ways of fusing LiDAR with a camera to obtain a 3D color model. Considering the convenience, the speed to fuse of two kinds of data and the volume of the procedure data, we use extrinsic rigid-body calibration to match the data of LiDAR and the camera. The three important devices mounted on a moving platform are 2D LiDAR sensor (Hokuyo UTM-30LX-EW LiDAR), digital camera (Nikon S3000) and robotic servo (Turret Widow X servo). Extrinsic calibration for LiDAR and camera is completed through getting rigid-body transformation. The relationship between the data of LiDAR sensor and the digital camera is used to calculate the transformation matrix. It requires us to translate the 2D camera coordinate frame to the 3D LiDAR sensor coordinate frame by using extrinsic transformation. Before fusing the digital camera image and the LiDAR data to construct a map, the key requisite is to gather data from two sensors of a common target. This means that the two sensors detect a fiducial target of a front view of the detecting system at the same time. A planar checkerboard pattern is usually used as a fiducial target, because the white and black patterns are equally square and detection by both LiDAR sensors and cameras is easy.

The LiDAR sensor obtains a 3D points cloud of a planar checkerboard in distance and intensity. The camera can get the color and texture data in two dimensions from the fiducial target checkerboard. It is hard to directly match the 3D points data with 2D pixels. We need to convert the data obtained from the LiDAR sensor of 3D points into 2D points in a plane, using the camera projection model. After converting the LiDAR data, the point-to-point correspondence between LiDAR 2D points and the camera data is found. We find the correspondence through extracting the features of edges and corners from the checkerboard. The checkerboard pattern from the camera is captured automatically in MATLAB. The checkerboard features in the LiDAR data is obtained manually, because the intensity information is not strong enough for automatic extraction. To realize the function of an omnidirectional camera, we set the system of the camera and LiDAR in three positions to get a wider front view. After getting the correspondence of LiDAR 2D points and the camera data, the points with color and texture need to be projected back to 3D coordinate, according to the sequence of the points projected from a 3D points cloud to a plane.

In this Chapter, it mainly describes the target-based extrinsic calibration. In Section 3.2, it describes the way to project a 3D points cloud into a plane with intensity information. In Section 3.3, it describe the method for getting the transformation matrix.

3.2 Intensity Image Generation

3.2.1 2D Representation of LiDAR Data

From Chapter 2, we know that 3D points are generated by combining LiDAR and servo as one system. Figure 3.1 shows the 3D points cloud colored by the intensity data that is captured from LiDAR for a fiducial target in our lab. After detecting the planar checkerboard pattern, we apply the LiDAR sensor to a hallway environment to get accurate and dense 3D data. A range sensor obtains a number of points in a 3D indoor

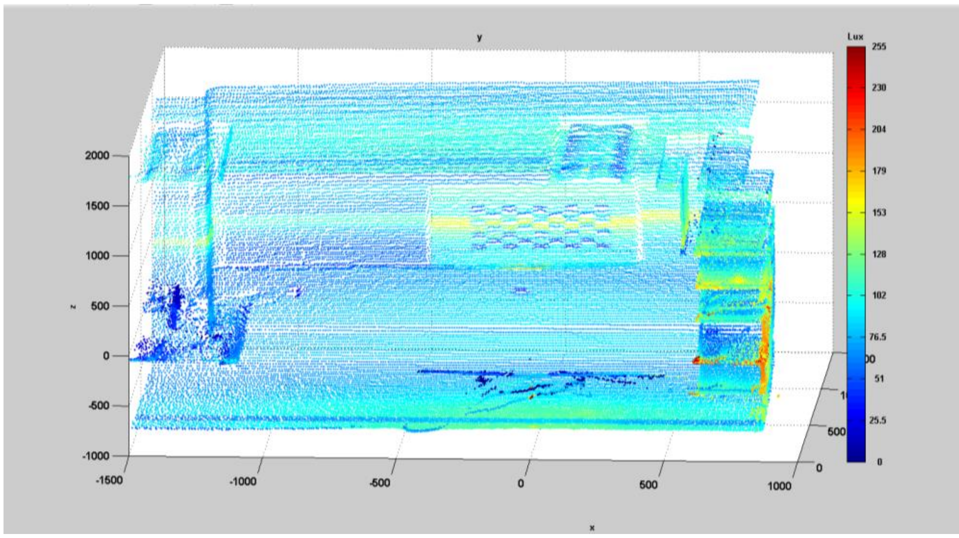


Figure 3.1: LiDAR 3D points cloud with intensity value.

environment by doing extrinsic calibration and registration. Then we put the data into the MATLAB to visualize the 3D points. The color bar is set to the largest color range, so that the comparisons among different materials for the front view are obvious.

To match 3D points cloud with 2D image, we first need to convert the 3D points cloud into a 2D intensity image. The 2D representation of LiDAR sensor for this thesis has two modules. The first module is applying the camera projection method and projecting the 3D points cloud into a plane. The second module is getting an intensity image from the 2D points by using interpolation. The intensity value is used to color the 2D intensity image the same as the projected image, according to the camera model, which is called pseudo image.

3.2.2 <u>Project LiDAR 3D Data onto a Plane</u>

According to the camera model, we project the 3D points cloud into a 2D plane. A list of 3D points projected in a plane with intensity value. After we get the LiDAR 2D points image of intensity image, it is necessary to convert the data to grid form for getting the features from the intensity image. The LiDAR 3D data is in the world coordinate, and we need to generate the correspondence between world coordinates and image coordinates. The coordinates system for camera projection is shown in Figure 3.2. The image plane is located along optic axis by a focal length of f units.

The principle of a pinhole camera projection is used for converting a list of LiDAR 3D points to a LiDAR 2D intensity image. The XYZ coordinates are used to the world coordinates. U_c and V_c are the coordinates of the points in world coordinates projected to a plane, and the plane is a focal length away along the original position (the camera lens). Z is the optic axis. The X Y plane parallels the focal length plane. We complete the perspective projection via the triangle rule. According to similar triangle

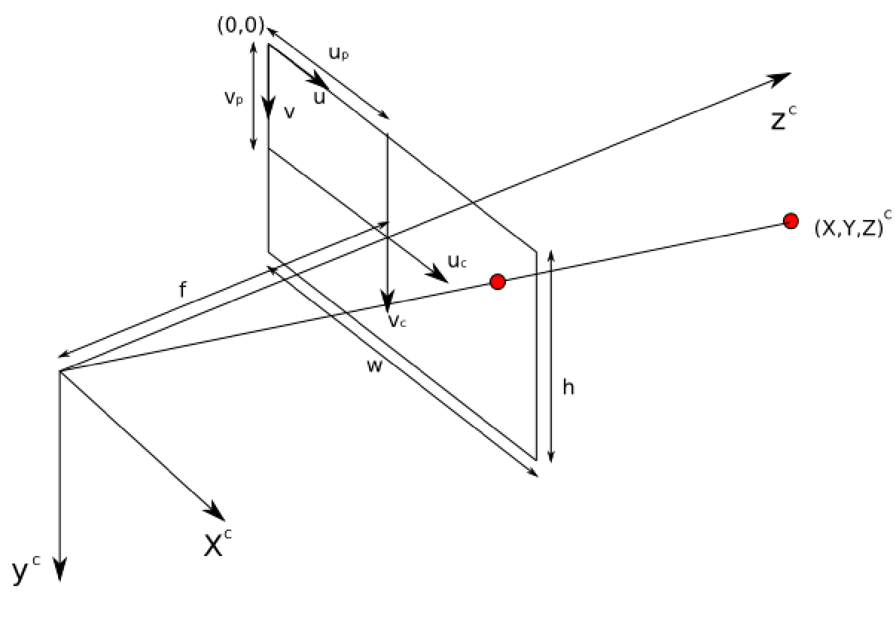


Figure 3.2: The principle of camera projection.

rules as shown, we can get the correspondence between the image coordinates and world coordinates.

The application of homogenous coordinates is to convert points from the world coordinates to camera image coordinates. For example, by adding a fictitious third coordinate w for existing coordinates x and y , the points represented from a 3D coordinate (x, y, z) to a 2D coordinate (u, v). And a point (u, v) in cartesian coordinates becomes (x, y, w) in homogeneous coordinates, which $u \; and \; v$ are re-expressed as $u = x/w$ and $v = y/w$. The equations (3-1) for the triangle rule as follows.

$$u = f\frac{x}{z}\ ; \qquad\qquad v = f\frac{y}{z}; \tag{3-1}$$

where (x, y, z) are the points in the 3D world coordinate, and (u, v) are the points in the 2D image coordinates, f is the focal length.

We apply co-linearity equations for projecting a 3D LiDAR points cloud onto a defined plane. Once we draw a line to connect the original point of camera coordinates with the points in the world coordinates, a conjunct point can be found in the focal length plane. Figure 3.3 shows the projected 2D points with intensity value. There is no direct match between LiDAR 2D points and camera image. We need to interpolate the intensity value of the points in the 2D world coordinate to an image in the 2D image coordinates.

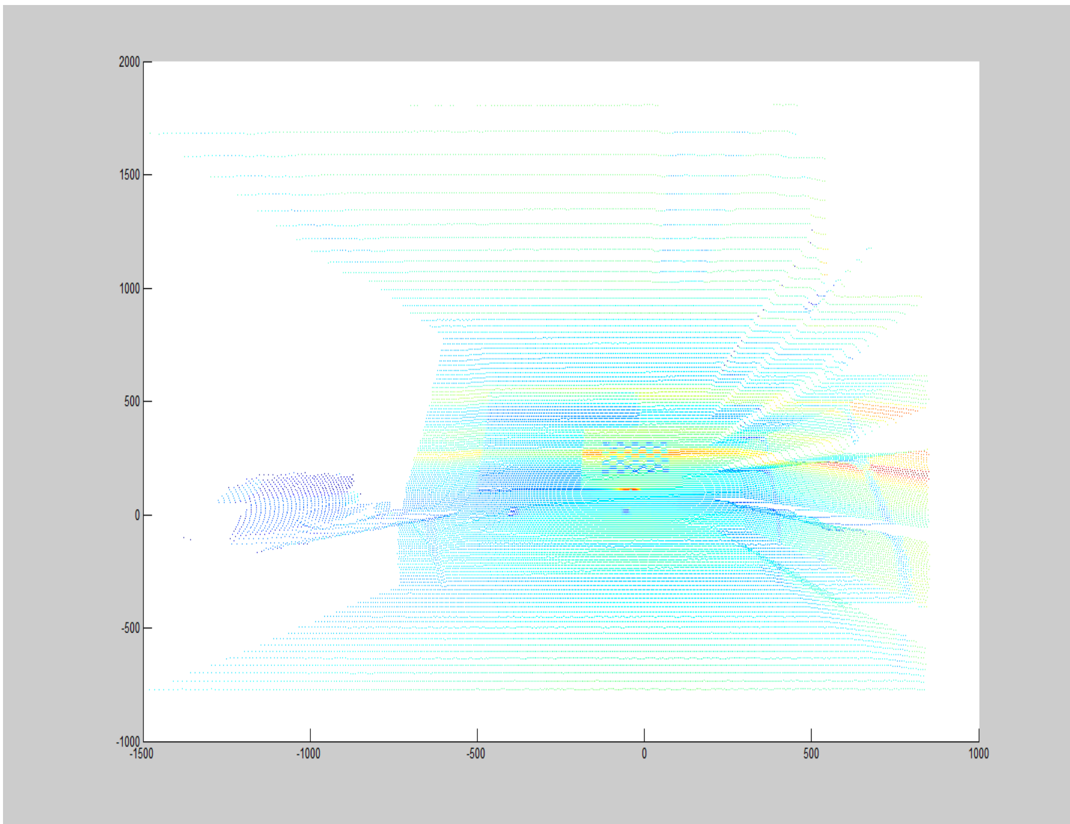


Figure 3.3: Projection of 3D points into 2D plane using the camera model.

3.2.3 Interpolation to an Intensity Image

We need to interpolate the LiDAR 2D points to a LiDAR intensity image. Interpolation is a procedure for taking a discrete set of data points that we know and estimating the value at new data points according to the value of the known data points. The LiDAR 2D points in the plane are not in evenly discrete format. So we need to predefine grids for interpolation. After defining the grids, we use the location and intensity value of the LiDAR 2D points and to locate the nearest data value and assign the same intensity value. The nearest neighbor interpolation is a convenient interpolation method because of its speed and simplicity. The intensity image is shown in Figure 3.4.

For the MATLAB code, before interpolation, we predefine the grids by using the MATLAB function $meshgrid$. After, getting the grids, we translated the LiDAR 2D

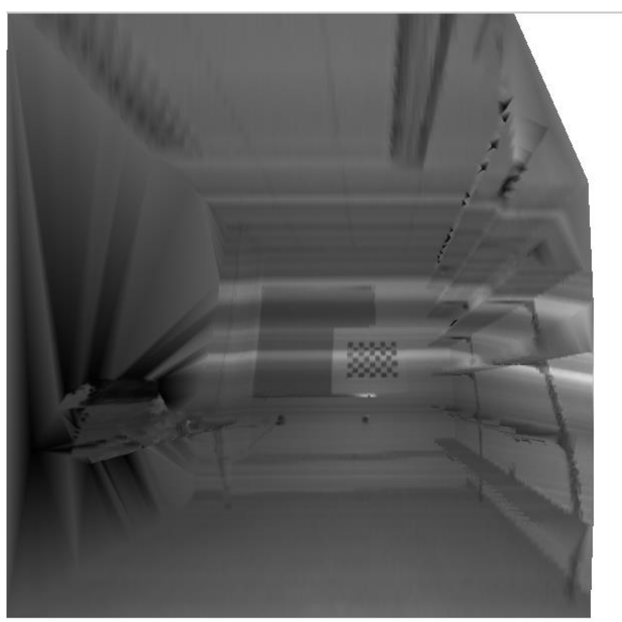


Figure 3.4: Intensity image obtained by interpolation.

points in the gridded image.

Once we translated the LiDAR 2D points in grid cell, we use the interpolation method to assign the intensity value to the grid. The nearest-neighbor interpolation method is used for finding the value of the grid by assigning the nearest integer neighbors with the same value of 2D LiDAR points. We use the $griddata$ function to interpolate the squares of the query points specified by (X, Y), and it can return the interpolated values (ZI). The 2D LiDAR points have no structure or order between their relative locations, we can also use triangulation to interpolate the scattered data. It will be easy to find the world coordinates for the point when using the even grid to interpolate instead of using the triangulation. Finally, a greyscale intensity image from LiDAR quantized between$[0, 255]$ is obtained. It will generate different results when the data is interpolated with different query points.

3.3 LiDAR-Camera Transformation Matrix Generation

3.3.1 Checkerboard Method

Extrinsic calibration is the key for fusing the data of the multi-sensor system. The convenient method for extrinsic calibration of the 2D LiDAR sensor and the camera is using a planar checkerboard as the fiducial target. A fiducial target is an object placed in the field of view of the imagery system. The fiducial target can be any object, such as a trihedral calibration rig, a polygonal board, a planar checkerboard or a single circle [3, 11, 21]. The most common fiducial target is a planar checkerboard pattern and it will be used in this research for extrinsic calibration. The checkerboard is used as the calibration target, and constructed constraints on the extrinsic parameters by creating data association between planes of the interpolated LiDAR intensity image and the stitched camera panorama. Because the checkerboard pattern is visible for both the LiDAR sensor and the camera, it is used to extrinsically calibrate a 2D LiDAR and a digital camera. The correspondence for extrinsic calibration between the LiDAR and the camera is a point-to-point correspondence. There is an advantage to using the planar checkerboard pattern as a fiducial target. The planar checkerboard is a board of checkered pattern. It consists of white and black squares. Features like the corners of the checkerboard are easy to extract and recognize by the detecting system. The errors of estimate extrinsic calibration parameters can be minimized by selecting a large number of points from the planar checkerboard pattern.

We put the checkerboard on the wall in front of the LiDAR-camera system, and obtain the data of the checkerboard from both sensors. The checkerboard pattern need to be captured from two types of images: the grayscale interpolated LiDAR intensity image and camera color panorama. For finding a transformation matrix to align a LiDAR 3D points cloud and a 2D image, the checkerboard needs to be captured simultaneously by the LiDAR sensor and the camera. The planar checkerboard pattern is used to extrinsically calibrate a LiDAR range scanner and a monocular camera system. To detect the checkerboard in a camera panorama as shown in the left image from Figure 3.6, we first select the region of the checkerboard from the whole panorama for easy detection. For the portion of the camera panorama, we use a MATLAB code to find the checkerboard patterns. Then we can get the size of each square of the checkerboard pattern and the location of the features of the checkerboard. The numbers and the positions of the corners of the checkerboard are obtained in MATLAB. Next, the information of the features of the checkerboard in the intensity image need to be extracted for fusing the multi-sensor data.

An intensity image is interpolated from the LiDAR 2D points and the checkerboard pattern is detected from the intensity image. We first select the region of the checkerboard from the LiDAR intensity image. The checkerboard in the whole intensity image is difficult to recognize and the intensity information can only produce a weak value. It difficult to recognize and extract the relatively small features of the checkerboard in the whole intensity image. Then we use the MATLAB code to highlight the corners of checkerboard for easy recognition. The corners of the checkerboard are marked with red spots. The checkerboard pattern in the LiDAR intensity image is shown in the right image from Figure 3.5. We use a mouse to manually obtain the information of the checkerboard pattern. The information includes the size of the checkerboard pattern, the size of the white and black squares of the checkerboard and the sequences and locations of the feature points of the checkerboard. We finally obtain the features of checkerboards from both the images.

For one point in the checkerboard pattern, we want to find the correspondence between that point in the LiDAR intensity image and in the camera panorama. The checkerboard pattern is used in this section as a common fiducial target [12, 22, 24, 27]. The checkerboard pattern in the previous section is used as an intrinsic calibration target to correct the distortion of the camera images.

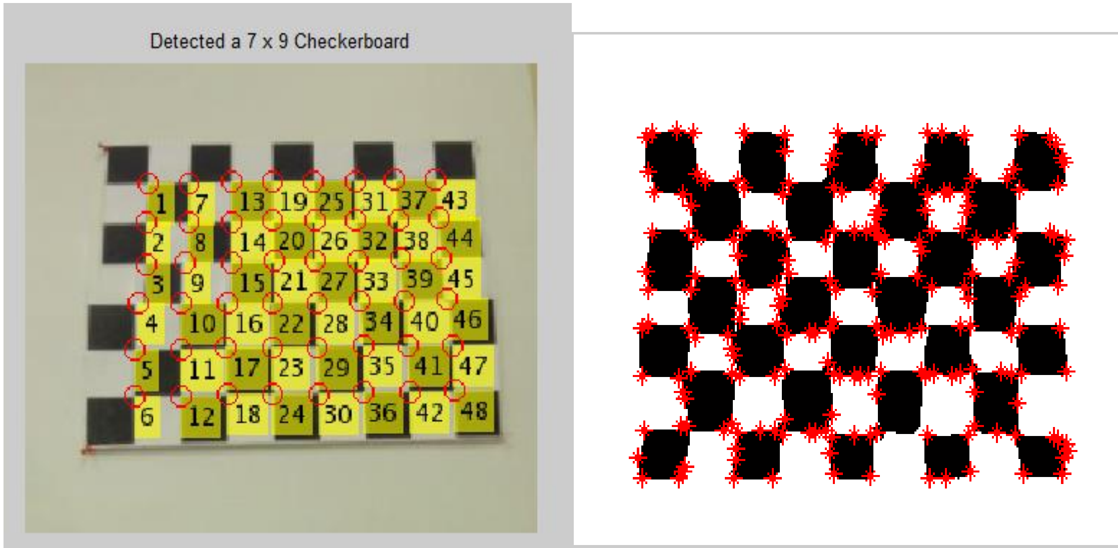


Figure 3.5: Checkerboard of panorama image (left) and of intensity image (right).

3.3.2 RANSAC Algorithm

Extrinsic calibration is a method to find the correspondence between the LiDAR 3D points cloud and the camera 2D image. The RANSAC algorithm is used to find the correspondence between the LiDAR intensity image and the camera panorama. The RANSAC algorithm is a hypothesis-verification algorithm based on a distance threshold. By using the RANSAC algorithm, the MATLAB $estimateGeometricTransform$ function outputs a minimal error solution. In the RANSAC algorithm, a pair of points p_i^a (LiDAR intensity image, imagePoints1) and p_i^b(panorama image, imagePoints2) is an inlier only when the distance between p_i^b and the projection of p_i^a based on the transformation matrix falls within the specified threshold. The distance metric equation (3-2) used in the RANSAC algorithm is as follows:

$$d = \sum_{i=1}^{Num} min(D\left(p_i^b, \Psi(p_i^a : H)\right), t) \quad (3\text{-}2)$$

where p_i^a is a point in the LiDAR intensity image; p_i^b is a point in a camera panorama; $\Psi(p_i^a : H)$ is the projection of a point on the intensity image based on the transformation matrix H; D is the distance between a pair of points; t is the threshold; and Num is the number of the points.

Given a corner of the checkerboard at position (x, y) in the LiDAR intensity image, the search for a match considers all features of the checkerboard within a region centered on (x, y) in a camera panorama with a threshold on maximum disparity. We measure the sum of squared differences in intensity for the value of possible matches. The threshold for matched points can maximize the possibility of correct matches.

The relation between a point in the LiDAR intensity image and the same point of checkerboard in panorama is projective. It works through repeatedly obtaining solutions estimated from a minimal set of correspondences. The correspondence are obtained from the data of the LiDAR intensity image and the camera panorama. Then, each solution is verified using the support from the complete set of putative correspondences to find the most fitting model.

There are three steps for the RANSAC algorithm. First, we need to randomly select a small sample subset to estimate a fitting model. Next, the elements that fit the estimated model will be considered as inliers; otherwise they are considered as outliers. We need to test each solution for support from the complete set of putative correspondence. Finally, we repeat generating solutions estimated from the minimal set of correspondences, which are gathered from the data and generate the transformation matrix.

3.3.3 Transformation Matrix Generation

The RANSAC algorithm is used for extrinsic calibration of the LiDAR range sensor and the camera. The LiDAR intensity image and the camera panorama captured from both sensors are the prerequisite for fusing the complementary information. The correspondences of the features of the checkerboard are used to establish geometric constraints for computing the extrinsic parameters. We want to find a rigid transformation between camera coordinates and LiDAR coordinates. Finding a transformation matrix between the camera panorama and the LiDAR 2D intensity image is the main problem for fusing the data of the two sensors. Point-to-point correspondences are used in my research. The calibration of the LiDAR sensor and the digital camera requires simultaneous observation of the checkerboard. We detect the features of the checkerboard from the LiDAR intensity image and camera image separately by selecting the size and corners of the checkerboard. The points manually selected from the intensity image, have point-to-point correspondences with the points automatically selected from the camera panorama to establish constraints on the extrinsic parameters. Because the camera is mounted on the top of the LiDAR, the correspondence is rigid for this system, which is called the transformation matrix. After importing the point pairs into MATLAB, we can get the transformation matrix of the LiDAR intensity image and camera panorama.

3.3.3.1 The Transformation System

The estimate geometric transformation block shown in Figure 3.6. can be used to generate a transformation matrix for the LiDAR sensor and the camera. We select the features of the checkerboard from the panorama as one set of input and also select the features of the same checkerboard from the intensity image as the second set of input. We need to select at least four pairs of points as inputs to find the transformation matrix. Here, we put n pairs of points into MATLAB and get the transformation matrix according to the projective relation between the two images.

For stitching the panorama to the intensity image, the geometric transformation panorama is one input image, and it is generated by applying the transformation matrix to the panorama. The LiDAR intensity image is the second input image. We use the MATLAB $imwarp$ function to complete the transformation of the panorama image. We can also use the same MATLAB function for stitching them together to get a transformed image. We use that function to apply geometric transformation to camera panorama, and

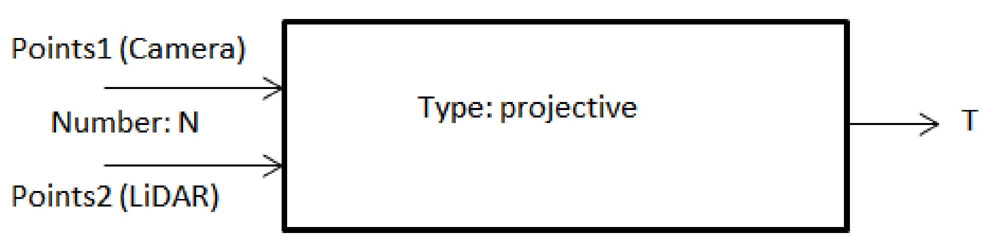


Figure 3.6: Estimate geometric transformation block.

the transformed panorama has the same image size as the intensity image. The checkerboard pattern in the camera coordinate frame can be estimated by the transformation matrix to the LiDAR intensity image frame. Finally, we get the intensity image matched with the panorama. The procedures are shown in Figure 3.7.

3.3.3.2 Transformation Matrix

According to the RANSAC algorithm, we can use this method to generate the transformation matrix. We need to get co-observable features of the checkerboard from sensor modalities of both the LiDAR data and the camera image. The checkerboard pattern is used because this pattern is easy to extract from both image data and LiDAR

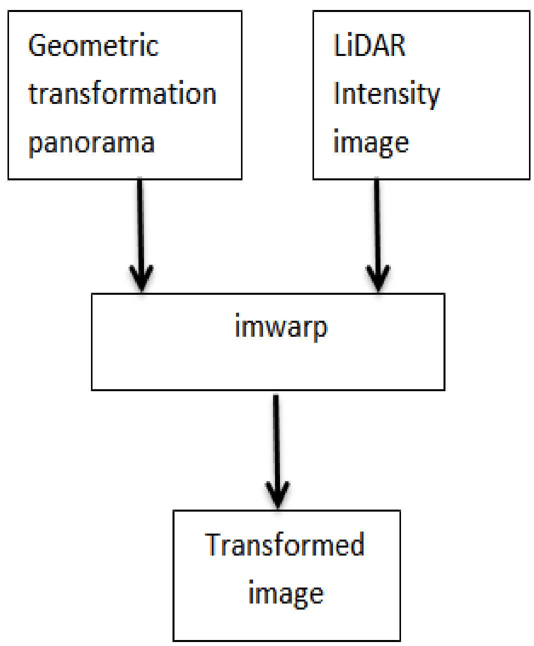


Figure 3.7: 2D geometric transformation generateion.

2D intensity data. In the experiments, we use a planar checkerboard pattern to complete extrinsic calibration by finding the correspondence between the two images. As shown in Figure 3.8, the perception sensors, Hokuyo UTM-30LX-EW 2D LiDAR scanner and digital camera, are used for generating the transformation matrix. The 2D LiDAR sensor generates an intensity image, and the digital camera captures the images of environments. The two sensors capture the locations of the same point on the checkerboard. Then the correspondence between the two sensors can be generated as the transformation matrix. A similar method can be used for the extrinsic calibration of a 3D LiDAR and an omnidirectional camera [7, 8].

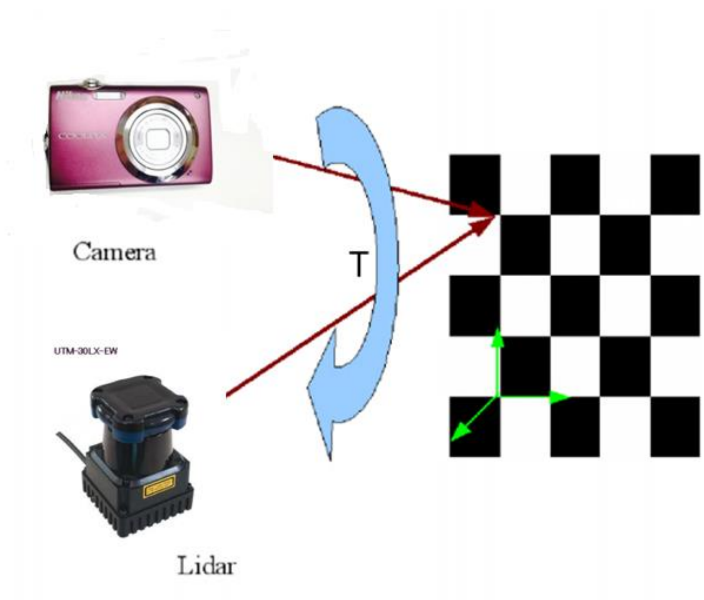


Figure 3.8: The experiment for extrinsic calibration of the 3D LiDAR with camera.

We align images that were taken from the same fiducial target with different sensors. A planar checkerboard pattern is set in front of the rigidly mounted camera and 2D LiDAR-servo multi-sensor system. The images are captured by the Nikon digital camera and distortion is corrected by using the MATLAB camera calibration app. The corrected images are stitched to the panorama by Panorama Maker 6 software. The images are in two-dimensional coordinates and we set a point of the image as (U, V). The distance and reflectivity information is obtained from the LiDAR sensor. The 3D points cloud is captured from the combined 2D LiDAR sensor and servo system and it belongs to three-dimensional coordinates. A point in the world coordinates is described as (X, Y, Z).

For the multi-sensor system, we can find the correspondence between homogeneous 3D points that are gotten from LiDAR coordinates $P_w = [x\ y\ z\ 1]^T$ and a projected point from camera image coordinates $P_c = [u\ v\ 1]^T$. The correspondence is shown in the following equation (3-3):

$$P_c = K[TR]P_w \tag{3-3}$$

where P_w represents the points in the world coordinates and P_c represents the points from the field in front of the camera which are projected into an image. K is the intrinsic matrix of the camera. T, R are the extrinsic matrix, which are related to the transformation from the world coordinate system to the camera coordinate system.

The most important problem here for calculating the correspondence between the world coordinate (X, Y, Z) gotten from LiDAR sensor-servo system and the camera coordinate $(U, V)$is to find transformation matrices of both intrinsic and extrinsic parameters. A point is transformed from 3D world coordinates to 2D image coordinates in two steps. By using the Nikon digital camera parameters, the points obtained from the LiDAR in world coordinate are projected into the 2D plane. The transformation matrix of a point in the LiDAR intensity image and the related point in the camera panorama need to be generated. The problem for projecting LiDAR 3D points onto a plane is previously solved. The parameter t in the above equation is the transformation matrix for converting the 3D points cloud of the LiDAR sensor into the 2D points. The focal length and intrinsic parameters of the pinhole camera are used for the camera projection of the 3D points cloud. The intrinsic parameters of the pinhole camera is used to correct the distortion of images. The focal length is used to project a list of 3D points to a plane according to the similar triangles rule.

After applying the camera model projection method, the problem becomes to matching 2D points in the intensity image with the camera image. The next step is to match the interpolated intensity image from the projected LiDAR 2D points with the camera panorama. The features of a checkerboard from both the LiDAR interpolated intensity image and the camera panorama image is used to estimate the transformation matrix. By adding a dark area that is not contained in panorama, we set the panorama to the same size with the intensity image. Then it will be easy to put the color and texture information back to a list of LiDAR 3D points. The size of the panorama and the size of the LiDAR intensity image are different. The ratio of these two images can be calculated by the size of the same checkerboard both in the panorama and in the intensity image. After applying the ratio, the size of the panorama image is converted to the size of the intensity image. Then a list of features of the checkerboard is used to find the extrinsic transformation matrix to map the data of panorama to the image which has the same size as the LiDAR intensity image. The important part for transformation is to find the transformation matrix T. By using transformation matrix T, the point from camera panorama coordinates (U, V) is projected to intensity coordinates (X', Y'). By finding the projective relationship between the projected intensity point $P_I = [x' \ y']^T$ with the point in camera image $P_c = [u \ v]^T$, the transformation matrix T is generated. The transformation equation (3-4) is as follows:

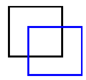
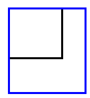
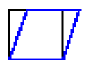
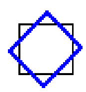
$$P_I = TP_c \tag{3-4}$$

where P_c is the point in camera image coordinates, P_I is the point in intensity image coordinates, and T is the transformation matrix that maps the point from camera image coordinates to intensity image coordinates [18].

As shown in Table 3.1, the transformation matrix is a 3-by-3 matrix. It is specified as a projective transformation and has 8 degrees of freedom. Projective transformation supports translation, rotation, isotropic scaling, and tilting. The projection of a point $[u \ \ v]$ by T is represented by homogeneous coordinates as $[x' \ \ y' \ \ w] = T[u \ \ v \ \ 1]$. The homogenous coordinates enable all projective operations to be expressed as a matrix multiplication.

The algebraic distance for a pair of points, $[x^a \ \ y^a]$ on the LiDAR intensity image, and $[x^b \ \ y^b]$ on the camera panorama, according to projective transformation T, is defined as equation (3-5):

Table 3.1: Transformation matrix for translation, scaling, shearing or rotation.

Affine Transform	**Example**	**Transformation Matrix**	
Translation		$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	t_x specifies the displacement along the *x* axis t_y specifies the displacement along the *y* axis.
Scale		$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	s_x specifies the scale factor along the *x* axis s_y specifies the scale factor along the *y* axis.
Shear		$\begin{bmatrix} 1 & sh_y & 0 \\ sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	sh_x specifies the shear factor along the *x* axis sh_y specifies the shear factor along the *y* axis.
Rotation		$\begin{bmatrix} \cos(q) & \sin(q) & 0 \\ -\sin(q) & \cos(q) & 0 \\ 0 & 0 & 1 \end{bmatrix}$	q specifies the angle of rotation.

$$D\left(p_i^b, \Psi(\mathrm{p}_\mathrm{i}^\mathrm{a} : \mathrm{T})\right) = ((x^a - w^a u^b)^2 + (y^a - w^a v^b)^2)^{\frac{1}{2}}. \qquad (3\text{-}5)$$

where $[x^a \;\; y^a \;\; w^a] = T[u^a \;\; v^a \;\; 1]$.

The MATLAB *imwarp* function is used to align the translated panorama to the related LiDAR intensity image. It first applies the transformation matrix to convert the camera panorama in an image that has the same size as the LiDAR intensity image. Then we map the pixel value from the camera panorama to the LiDAR intensity image.

In Figure 3.9, the axes in the input space of the panorama are labeled by u and v , the axes in output space of the transformed panorama are labelled by x and y. In the checkerboard pattern, each black, white, and grey square has the same length and width.

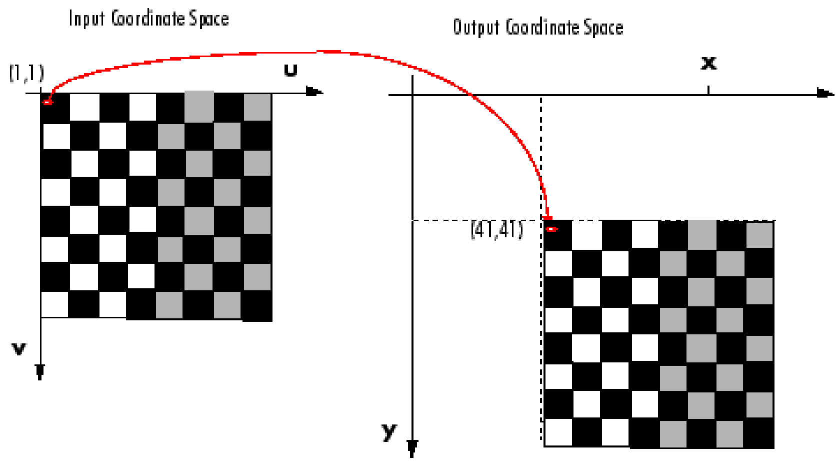


Figure 3.9: The input image after translated.

It means when we transform the value of the RGB image (which means color and texture information), we can specify different fill values for each RGB image.

3.3.3.3 Complete Transformation in MATLAB

We automatically capture the features of the checkerboard such as the corners from the camera panorama by using the MATLAB function *detectCheckerboardPoints*. This function is used for detecting black and white squares of the checkerboard from the 2D color image which was obtained from a common digital camera. We manually select the corners of the checkerboard from the LiDAR intensity image. We select part of the images of the checkerboard from the intensity image and the camera panorama to select the features easily. Then we make the grayscale intensity image more recognizable by changing the grayscale to black and white. Next, the corners of checkerboard are marked with red spots in MATLAB. Even though the corners are marked with red spots, they are not accurate enough for the features of the checkerboard. It is easy for us to select the points by hand after marking the features. After marking the features, the points are easier to select by hand. We select the features by hand in order. The sequence of the selected points of the LiDAR intensity image and the number of the corners of the checkerboard automatically captured from the camera panorama must be the same. After selecting two sets of feature point pairs as input from the LiDAR intensity image and the panorama, the MATLAB function $estimateGeometricTransform$ is used to generate the geometric transformation matrix. This function will return the geometric transformation matrix. The function used in MATLAB is as follows,

$$tform = estimateGeometricTransform(matchedPoints1, matchedPoints2, transformType)$$

It returns a 3-by-3 transformation matrix-$tform$ and it belongs to the projective transformation. We take the feature points from the LiDAR intensity image as the first set of input-matchedPoints1. Then we take the feature points from the camera panorama as the second set of input-matchedPoints2. The matrix $tform$ maps from the inliers in matchedPoints1 to the inliers in matchedPoints2. We choose ‘projective’ as the transformation relation from three types of transformation relations. The other two relations are ‘similarity’ and ‘affine’ and they have less transformation relation types than the projective transformation. The minimum number of matched point pairs for the ‘projected’ transformation relation type is four, the other relation types need two and three, respectively. This is because the greater number of matched point pairs we have, the more accurate results for estimated transformation we will have. The most accurate transformation relation ‘Projective’ is chosen as the input in my research.

We take a large number of selected feature points in the LiDAR intensity image and repeat the step to choose generate transformation matrix. Then we can minimize the error for selecting the features of the intensity image. The MATLAB function $estimateGeometricTransform$ uses an algorithm to classify points into either an inlier point or an outlier point by a calculated threshold. This function uses the M-estimator Sample Consensus (MSAC) algorithm to remove outliers. This algorithm is similar to the Random Sample Consensus (RANSAC) algorithm. The RANSAC algorithm enables us to find the most fitting transformation model from a set of correspondences. The RANSAC algorithm we used here has been described in the previous section.

We estimate an equation to fit most of the projected LiDAR points of the checkerboard by using the RANSAC method. In this method, we first need to create a bounding region. This region includes all the target points of the checkerboard from the intensity image. Then the target LiDAR points $\{P_I^i; i = 1,2, \dots, N\}$ needs to fit the model which is calculated by the RANSAC algorithm. It means we get a list of best fit as potential LiDAR points input. There are many different transformation relations for geometric transformation. My research includes more than one type of transformations such as translation, rotation, scaling and tilting. There are more than three non-collinear points of each image which are selected as inputs.

3.3.4 Omnidirectional Image

3.3.4.1 Omnidirectional Image

Previously, our work is focused on one transformation matrix for extrinsic calibration of a laser range sensor and an optical camera. The multi-sensor system only captures the data with the digital camera in the middle position. We use the checkerboard to estimate the rigid-body transformation between the LiDAR intensity image and the camera panorama. The transformation matrix can allow the frame of the camera image to project to the intensity image which is generated from the LiDAR range sensor. We use intrinsic parameters gotten from the MATLAB Camera Calibration App to calibrate the images which are obtained from the individual digital camera. One problem here is that the size of the panorama image that is captured in the middle position for matching is too narrow. So, we need to get a wider view of the panorama for fusing with the LiDAR intensity image. According to an omnidirectional camera, we need to capture more panoramas to fuse with the intensity image. The Nikon S3000 common digital camera only captures a narrow view. We need to stitch camera images to a panorama and stitch panoramas to an image which has a wider view. In my research, there are three camera capturing positions in the horizontal plane. The rigid-body transformation matrices of the three camera positions are generated by the same method. It should be noted that we use one single camera for three camera positions to simulate an omnidirectional camera.

3.3.4.2 Three Different Camera Positions

According to the principles of the omnidirectional camera, we need to complete extrinsic parameter estimation of the camera image corresponding to the LiDAR 2D intensity image. Because the common digital camera has a limited view, we set three measured poses for the single digital camera. In one of the camera poses, the camera head is put in the middle position and faced in front of the camera and LiDAR system. For the other two poses, the head of the digital camera is 35^o rotated to the left side and the right side away from the original position.

The planar checkerboard pattern is captured in three poses for the single camera. Then we calculate the correspondences between the digital camera and the LiDAR sensor. A similar method is used for generating transformation matrices of the camera in the left side and in the right side. The LiDAR sensor keeps facing the field in front of the LiDAR-camera system, because we need the 3D points cloud from the front view for fusion. The three transformation matrices are used for stitching a panorama with the intensity image. For the left transformation matrix, the camera is rotated 35^o horizontally away from the principle axis to the left and the checkerboard is put at the left side. The left transformation matrix is used for stitching the left side panorama with the LiDAR intensity image. The right transformation matrix is generated by rotating the camera 35^o away from the principle axis to the left and the checkerboard is put at the right side. Figure 3.10 shows the images for the digital camera at the right position. The RANSAC algorithm is used for generating the three transformation matrices. For setting the three positions for the camera, the collected view in front of the camera and the LiDAR sensor system covers less than 180^o. In the experiment, each panorama image is generated by each camera position and at the same time the LiDAR sensor captures the same full 180^o view for the three camera positions.

3.3.4.3 Stitching Images

The transformation matrix is used to stitch the panorama to the LiDAR intensity image. The three rigid-body transformation matrices are applied to stitch the three panoramas with the same intensity image. Before we create the panorama, we use the intrinsic camera matrix from MATLAB Camera Calibration App to calibrate the image,

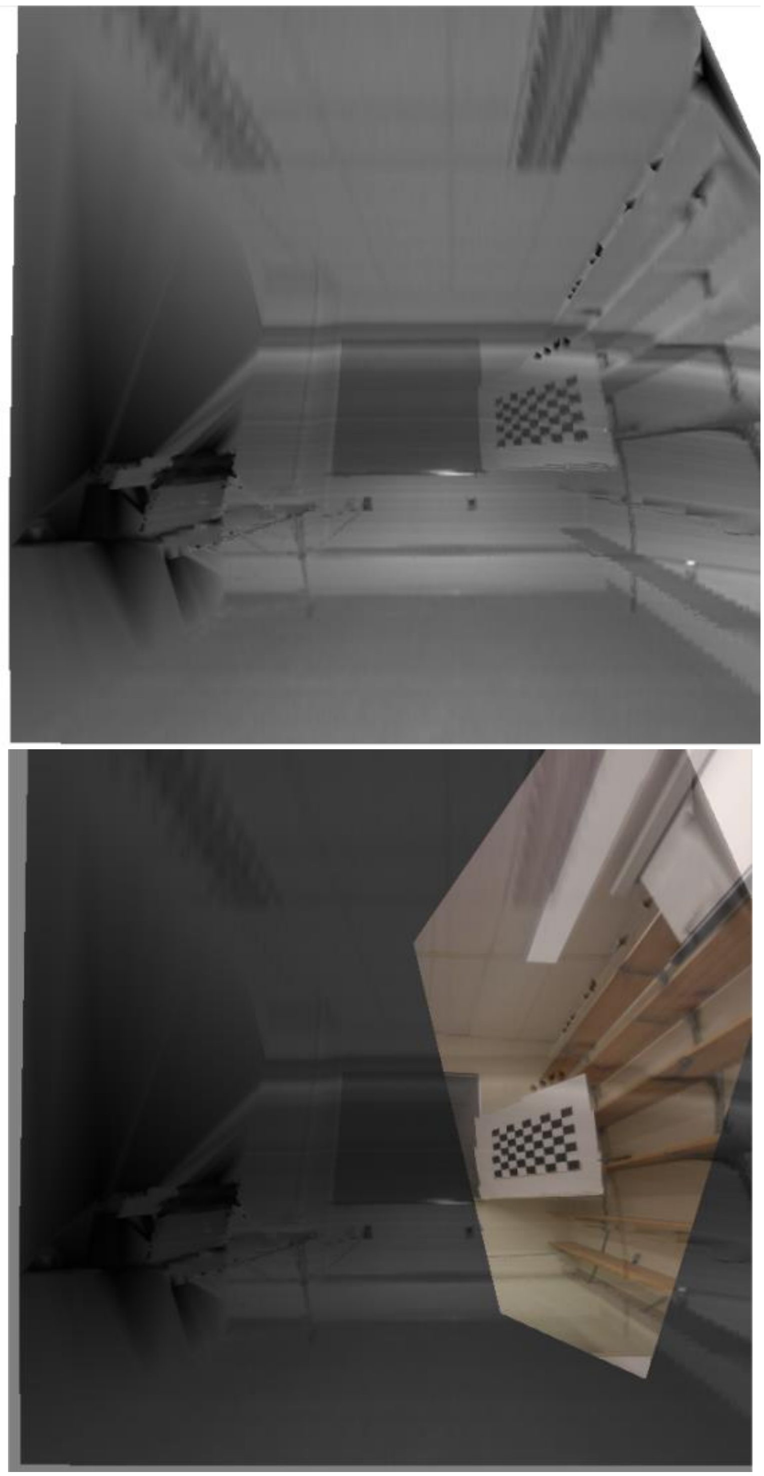


Figure 3.10: The intensity image of LiDAR data for camera in right position (top); matching panorama image and intensity image for camera in right position (bottom).

correcting the distortion of images. The camera sweeps the scene for the three camera positions from left to right along the horizontal plane. Then the three panoramas are generated for the front view. All the points on the panorama are applied by the transformation matrix $tform$ to map on the LiDAR intensity image. For generating the three transformation matrices, there are three sets of points by capturing the features of the checkerboard from the LiDAR intensity image and the panorama. All three panoramas are mapped on the LiDAR intensity image, and the size of the three transformed panoramas should be the same as the size of the LiDAR intensity image. In MATLAB, we use $imwarp$ function to map the three panoramas onto the same LiDAR intensity image.

Function $Vision.AlphaBlender$ is used to combine images, overlay images, or highlight selected pixels. Function $vision.AlphaBlender$ is used here for overlaying the three panoramas. Because the middle position transformation matrix has more accurate result for matching the panorama with LiDAR intensity image, the middle panorama is put on the top of these three panoramas. To obtain the best result, we first stitch the left side panorama and right side panorama with the intensity image. Then we overlay the middle image to the intensity image which has already been combined with two panoramas.

After stitching the three panoramas, the stitched intensity image is as shown in Figure 3.11. There are some errors when the intensity image was stitched with three combined panoramas. Because each match of the LiDAR intensity image and the camera panorama has errors, the combination for the three positions will also have errors. There

Figure 3.11: Stitching 3 positions of camera images.

is an illumination problem for the three panoramas because the camera captures the image in different locations. Although the three panoramas have different illumination problems in the research, we can still detect the features of the environment. The procedure for image stitching is the pre-required step for image registration. Registration of the textured 3D points will be described in the next chapter.

CHAPTER 4

3D MAP GENERATION

4.1 Image Registration

4.1.1 Intensity-Based Registration

Image registration in our research is the process of aligning the value of LiDAR intensity image with camera pixels. The intensity image is treated as the reference. After applying geometric transformation matrices to the panoramas, the camera image can be aligned with the reference image. The transformation matrix is obtained from matching the checkerboard of both the LiDAR intensity image and the camera panorama. The accuracy of the rigid extrinsic transformation matrix has great impact to the image registration result. Intensity based image registration maps the pixels of panorama images to the corresponding locations in the intensity image.

4.1.2 Image Locations

In this section, we describe the detail of camera image. Using pixel indices is one way to describe locations in an image. The coordinates of the pixel image are described from top to bottom and left to right. The image is on a grid of discrete elements. In pixel indices, the rows increase as they go downward and the columns increase as they go to the right. The pixel values are integer values on the grid. The pixel indices of an image and the subscripts of the two dimension matrix have one-to-one correspondence. Because of the one-to-one correspondence between the pixel indices and the subscripts for the same image, the relationship between pixel matrix and the way to display an image is easy to find.

Using spatial coordinates is the other way to express an image, it uses a continuously varying coordinates system. The spatial coordinate system for an image has correspondence with the pixel indices for an image. It includes the intrinsic coordinates and the world coordinates.

The pixels on an image are described by discrete elements and the intrinsic coordinates (x, y) describes the points of on an image. Functions in MATLAB use spatial coordinates and use pixel indices to describe the image. For one point in intrinsic coordinates, it can be located by specific column (x) and row (y). In an intrinsic coordinate system, the upper left corner of an image starts at $(0.5, 0.5)$ not the original point $(0, 0)$. In comparison, the upper left pixel is started at location $(1, 1)$ in the pixel indices. The intrinsic coordinate system has correspondence with the pixel indices. In my research, the intrinsic coordinate is used to specify the pixel of a point in world coordinate from pixel indices.

4.1.3 Color Intensity Image

We want to generate a realistic color 3D map. To accomplish that, we need to get visual data captured by a single camera, such as color and texture of the environment. The data will be precisely mapped back to the 3D points cloud by using the transformation matrix and registration method. Previously, we use transformation matrix to transform and stitch the panorama with the LiDAR intensity image. Registration of the color information with the intensity image is done by using the correspondence between world coordinates and pixel indices. Firstly, we use world coordinates to describe an image. It has certain X_{Data} and Y_{Data} properties for the image. In world coordinates, we can set the range of X_{Data} and Y_{Data} image properties different from that in intrinsic coordinates. After matching the intensity image and color panorama, each pixel in the intensity image is assumed to match with the pixel in the color image at the same location. The LiDAR 2D point has an associated $x\ and\ y$ image coordinates in the intensity image. Here the relationship between the pixel coordinate and world coordinate is found. Figure 4.1 shows the correspondence between pixel indices and spatial coordinates. The position of each point in intensity image is known and the point is generated by projecting 3D points cloud to 2D image. These positions are not necessary

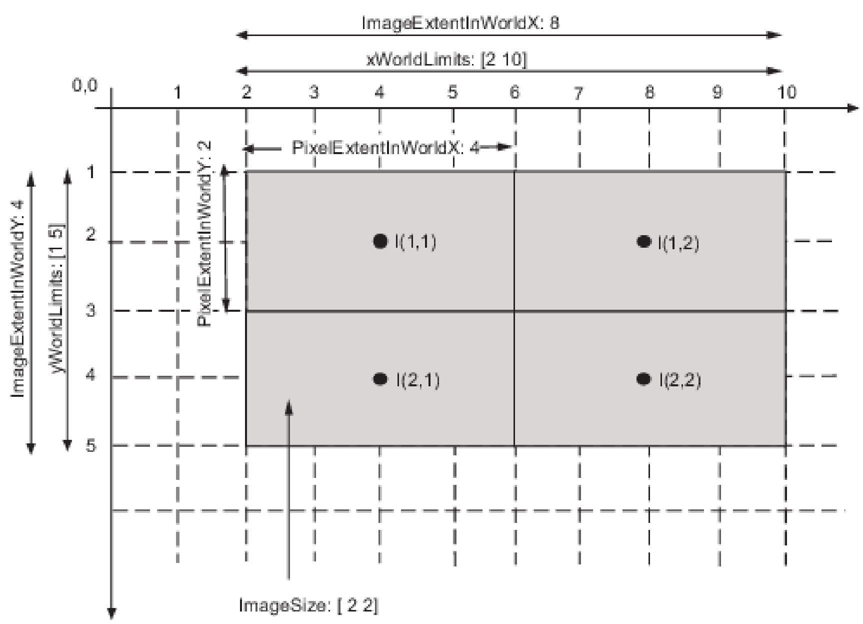


Figure 4.1: The properties in the elements of an image.
(http://www.mathworks.com/help/images/image-coordinate-systems.html)

integers and are unevenly distributed in the intensity image. In contrast, the pixel positions from panorama are integers and in square element format. We use $ceil$ function to get the nearest integer for the spatial coordinates of LiDAR 2D points. After getting the nearest integer for each point in the LiDAR intensity image, we can register the pixel value from panorama to each LiDAR 2D point. Using the correspondence between the pixel value in the panorama image and locations in world coordinates, we can make color registration and give the RGB value to each point of the intensity image. Then points with color and texture information are ready for next step.

4.1.4 Back Projection of 2D Points to 3D Points

Matching the camera panorama with the intensity image means we find the correspondence between pixel coordinates and spatial coordinates. After aligning intensity image with panorama, the color and texture information is registered on the intensity image, as shown in Figure 4.2.

Figure 4.2: Fusion intensity image with panorama image.

The intensity image corresponds to the LiDAR 2D points, which are the points projected from the 3D points cloud and interpolated to an intensity image. It is possible to color a 3D points cloud from an intensity image using a transformation matrix. The transformation matrix is used to get extrinsic calibration results and match the Lidar intensity with the panorama image. Then the nearest interpolation method is used to give the RGB value to each LiDAR 2D point in the intensity image. The sequence of the 2D points and 3D points cloud have been determined in MATLAB when we project the 3D points to 2D points, we can use the same order to give the RGB value from the 2D points back to the 3D points cloud. After getting the color and texture information, we import six parameters includes world coordinates X, Y, and Z and color information R, G, and B into MeshLab to generate 3D map.

4.2 Record in Dodge Hall

4.2.1 2D LiDAR and Digital Camera

After getting the transformation matrices from the fiducial target scene, we apply the target-based method to the data obtained from Dodge Hall Building in Oakland University. For one measuring location, the 3D points cloud is shown in Figure 4.3. The 3D points cloud with intensity value is shown in Figure 4.4. After projecting the 3D points to a 2D plane, as shown in Figure 4.5, the 2D intensity image is generated. Interpolating the LiDAR 2D points can generate the intensity image as shown in Figure 4.6. Because the camera is rigidly mounted on the top of the LiDAR, the transformation matrices are rigid. The three transformation matrices also can be used to align three camera panoramas with LiDAR 3D points for each measuring location in the hallway.

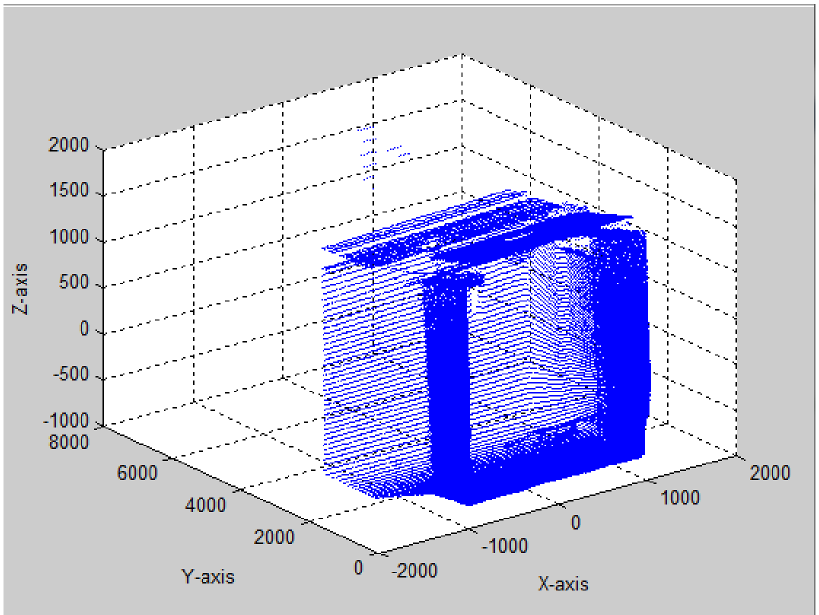


Figure 4.3: 3D points cloud obtained from LiDAR sensor in hallway.

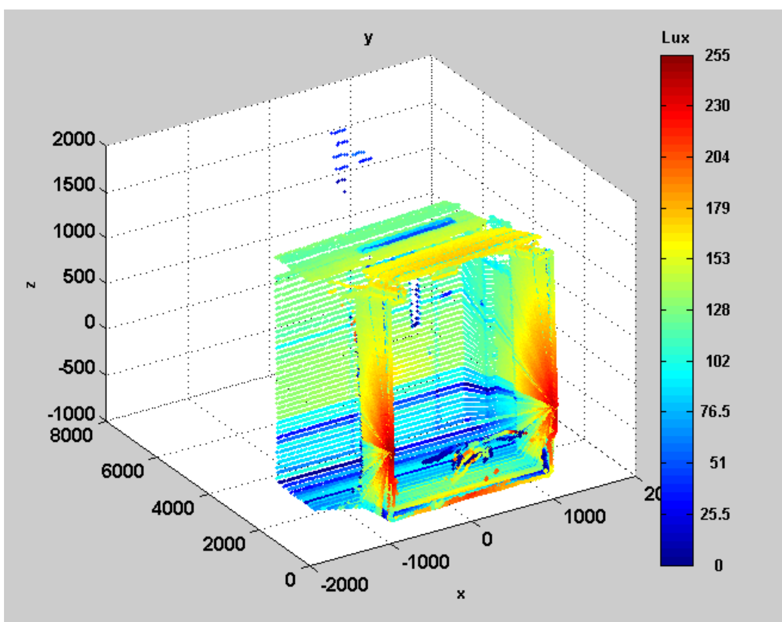


Figure 4.4: 3D points cloud with intensity image of location 1 in hallway.

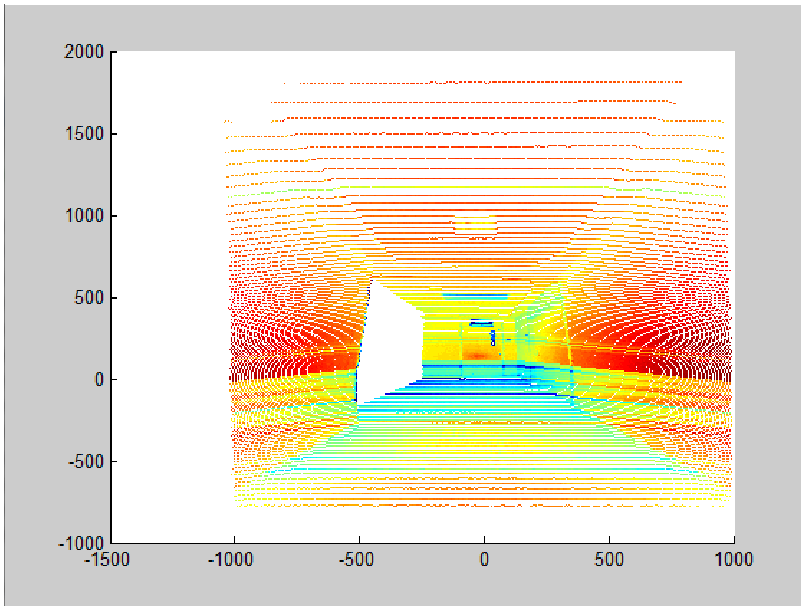


Figure 4.5: Projection of 3D points in a plane using a camera model.

Figure 4.6: Generation of LiDAR 2D intensity image for location 1.

Although the captured data of the camera images and the 3D points of the LiDAR sensor for the hallway are huge the three extrinsic calibrate matrices are the same and they can complete extrinsic calibration of the hallway data with less complexity. The data storage for the three transformation matrices only a little compared to the data storage for the image data of the hallway. We can use this method for generating a 3D map. Aligning the three panoramas with one intensity image at a measuring location in Dodge Hall, we can get the transformed image shown in Figure 4.7.

The procedure to get the textured 3D points for each location in the hallway is the same for fusing 3D LiDAR points with the panorama image in the fiducial target

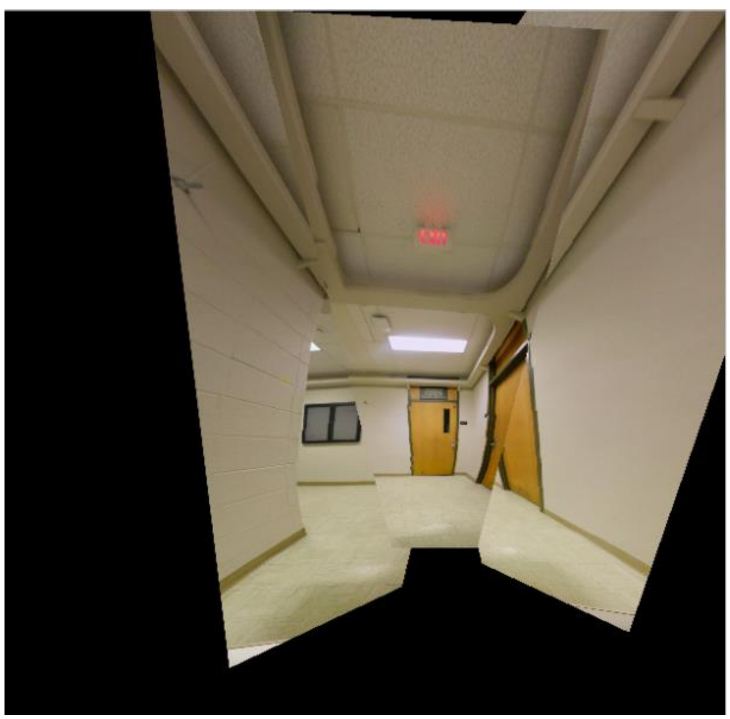


Figure 4.7: The combination of three panoramas in intensity image for one location in hallway.

situation. After getting the transformation matrices from the target-based checkerboard, we apply rigid-body extrinsic calibration to calibrate the hallway scenes by using the same three transformation matrices. As shown in the figures above, the distance and intensity information obtained by LiDAR is store in a 3D points cloud for each location in the hallway. The 3D points cloud with intensity information is projected onto a plane using a camera projection method. The extrinsic calibration parameters are used to align data of both projected 2D points and camera panorama for the hallway. After finding the correspondence between LiDAR scans and camera images for the hallway measuring locations, the 2D points are put back to 3D points using the sequence of projected 3D points cloud to 2D points. We then inport the 3D points with coordinates and color into MeshLab to generate a 3D model for one scene in the hallway.

4.2.2 Data Combination at Different Locations

The same three transformation matrices are used for the rest of the hallway data. Previous work is only for fusing images with LiDAR 3D points for one measuring location in the hallway. The data of different scenes are combined into the framework of a single hallway. There are four hallways scanned for the Dodge Hall Building, which is a small part of the whole. For each hallway, we align the detecting data using the same method described above.

All the data are measured in an indoor environment. A simple mobile platform is used to move the measurement system. The sample locations are measured and marked before scanning, making it more effective to do the experiment. Because of the limit of the range of the LiDAR sensor, it imposes limitation to the distance between two adjacent

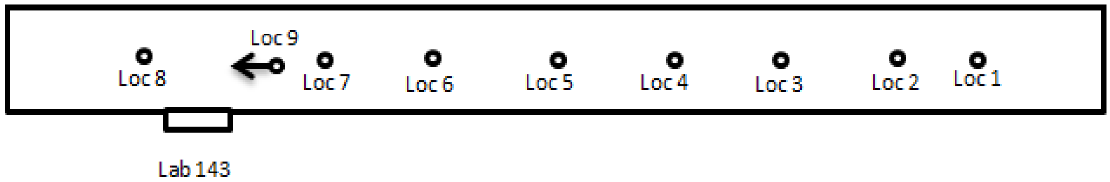


Figure 4.8: All detect locations for one hallway.

measuring locations. The number of scenes depends on the length of the hallway and the complexity of the environment. For indoor measurement, we need to generate a closed region by scanning from two opposite directions. As is shown in Figure 4.8, the direction for the LiDAR and camera system is to the right for locations from 1 to 8. The scan system faced the opposite direction, -i.e. left, for location 9. This allows us to generate data for a closed 3D model of the hallway, so which has no blank region in the image.

We align location 1 and 2 as a test for match of transformation matrices for LiDAR points and camera data. The distance between the first two measuring locations is 100 inches. The distance is measured in inches instead of in millimeter because it is the same as the unit of measurement on the ruler. According to the results of aligning the first two measuring locations, the error will be less if we set the distance of two adjacent locations be 150 inches. The distance of the first two measuring locations is too short, because we just use a few part of the textured 3D points cloud for alignment. The distance between two adjacent locations in one hallway is set as 150 inches. Only the length of first and second measuring location is different.

To merge different measuring locations in one hallway, a function $aliglocs.m$ is written for generating textured 3D points of the hallway. The function, has five inputs:

the data of current locations, the data at the second location, the distance between the two locations, the name for aligned data in $.txt$ format and the name for aligned data in $.mat$ format. After the date of two locations are merged, we treat the output data as the current system. Then, data at new locations can be added sequentially. To add the data at the last opposite directional location is added, we first need to mirror reflect the data and add it to the rest of the data for complete model of the hallway. During the procedure of merge, the region of the data for each detecting location needs to be selected because some points don’t have related color information. For two adjacent locations in a hallway, we set the length of the overlay region a little larger than the focal length of Nikon camera. The measured width and length of the hallway is used to select the region for the hallway. The models of other three hallways were generated using the same method.

4.2.3 Data Sampling locations in Dodge Hall

The sampling locations in Dodge Hall is shown in Figure 4.9. The measured

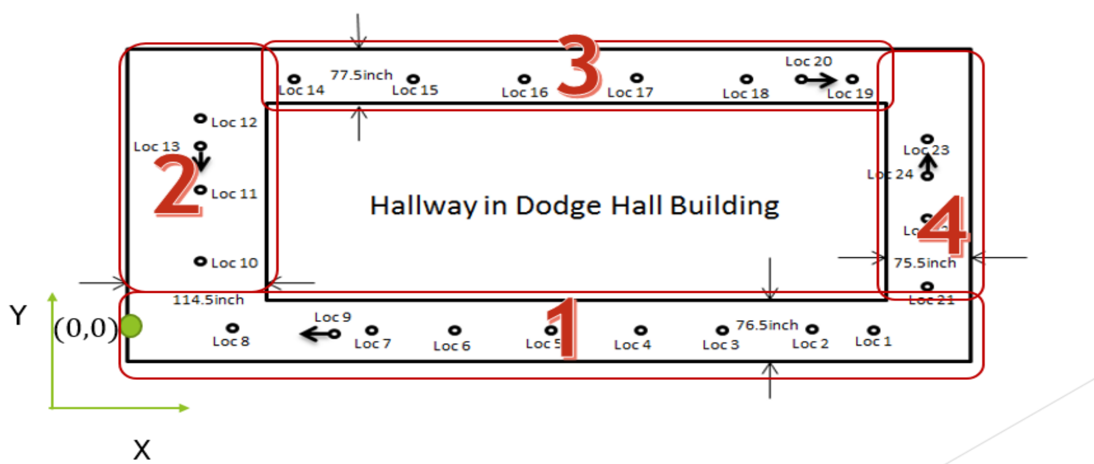


Figure 4.9: The map with sampling locations in Dodge Hall.

information includes the width and length of the hallway and the distance between two adjacent measuring locations [4, 20, 31]. There are a total of 25 measuring locations for mapping the Dodge Hall. The amount of data that we needed to collect is a relatively large.

The directions and the origin are necessary to describe the map for Dodge Hall. The origin of the map is set in the bottom left corner of the map. It is the same origin for the first hallway. After generating the maps of the four hallways separately, each hallway was assigned a selected region for connection. We selected the region of each hallway to avoid overlay during alignment. The second and fourth hallways have different directions than the first and third hallways. They must be rotated by 90^o. The other three hallways have a different origin than the first hallway. So they are translated to make a closed 3D model of Dodge Hall. The translation parameters were obtained from measurements in the experiment. There are four hallways for Dodge Hall at Oakland University. We need to align the four hallways to generate a 3D model for Dodge Hall. There are three steps for aligning four hallways including selecting, shifting and rotating. The region for each hallway in MATLAB is selected to avoid the repetition of two adjacent hallways. Because the original point for each hallway is starting from point (0, 0), a measured distance is shifted for putting the four hallways in one map. Again we need to rotate 90^o for the second and forth hallway.

4.3 3D Mesh Generation

4.3.1 MeshLab

MeshLab is an open source, portable, and extensible system for the programming and editing of unstructured 3D triangular meshes. The system is aimed to help the processing of the typical not-so-small unstructured models should in 3D scanning. The Meshlab software provides a set of tools for editing, cleaning, healing, inspecting, rendering, and converting this kind of mesh.

4.3.2 MeshLab for Extrinsic Calibration

MeshLab version1.3.3 is used for my research. It can import not only world coordinate, but also color value. Then we can generate a 3D points cloud with color and texture information received from camera panorama.

After importing the world coordinates data and texture data in $.txt$ format, MeshLab can complete information visualization. The match of the LiDAR 3D points cloud and camera panorama for one camera position is shown in the top of Figure 4.10. Using the MeshLab software, we can find that the camera image, and the 3D points cloud are matched well. We cannot display textured 3D points cloud in MATLAB. The bottom image from Figure 4.10 shows the combined digital camera panorama of three camera positions. The range of the black region, which indicates the points without color information, is the region between the head of camera and the focal length plane. Considering the principle of camera projection, the field of view at a distance less than focal length cannot be projected into the camera image plane. The black points should not be shown in the hallway for Dodge Hall. The overlap region between two adjacent

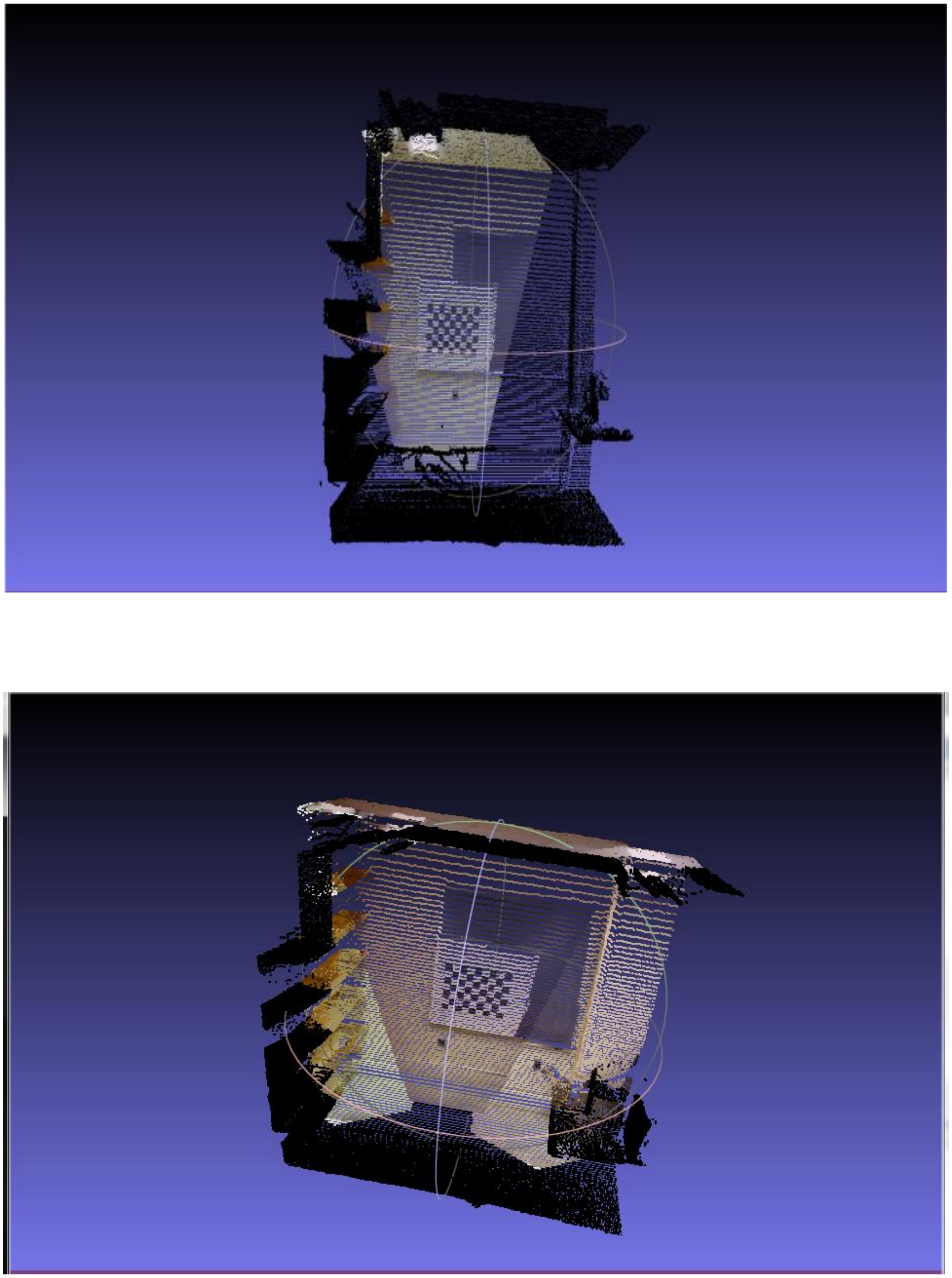


Figure 4.10: The Match of 3D Points Cloud and Panorama for One Camera Position (top); The Match of 3D Points Cloud and Three Panorama (bottom).

locations is used to remove the black region. We set the length of the overlap region a little larger than the length of the black region.

4.3.3 Experimental Results for Dodge Hall

We use the checkerboard to complete extrinsic calibration of the LiDAR sensor and digital camera at lab. After matching the LiDAR 3D points cloud with a camera panorama, the same transformation matrices for three camera positions are used in the hallway of Dodge Hall at Oakland University. We took 25 total sampling locations in the hallway of Dodge Hall. For each location, the same transformation matrix is used to align the 3D points cloud and the color image. Figure 4.11 shows the data in MeshLab for the first location. There are two black regions. One is between the center of the LiDAR-camera system and measuring objects, which is less than focal length. The other is around the mobile platform, which we use to move the multi-sensor system from one location to another in the experiment.

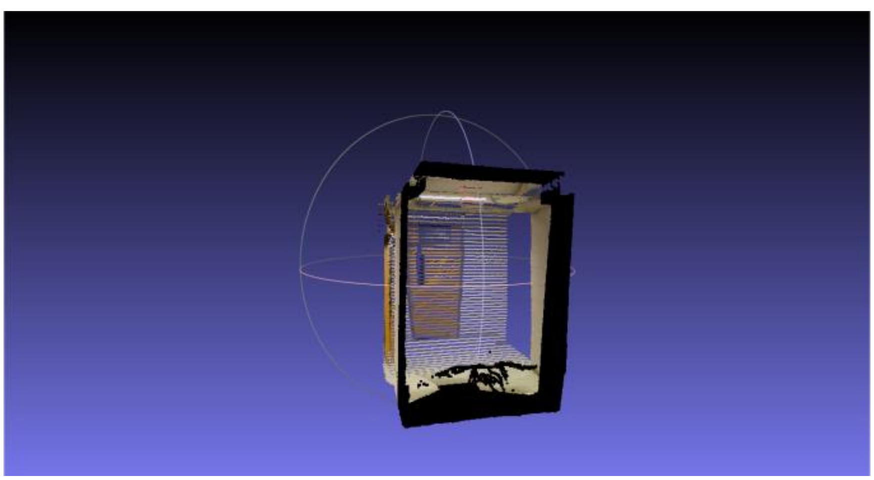


Figure 4.11: Textured 3D points in MeshLab for one location.

After getting the textured 3D points for each location, we combine the data of two adjacent locations in MATLAB. The black region is removed because there is an overlay between the data of the two detecting locations, and we set the distance to 600 mm for the length of the overlay in MATLAB for function $aliglocs$. We combine the data of two adjacent locations and treat them as a system. Keep adding the data of new locations to the current system, we can the data for the whole hallway. In Figure 4.12, the upper image is a 3D model for one hallway, and the bottom image is the inside view, portraying realistic colors for that hallway.

The 3D model for Dodge Hall is shown in Figure 4.13. We can tell the ceiling, lights and so on from the textured 3D points in MeshLab. The original point $(0, 0)$ is at the right upper corner in Figure 4.13. There are some blank regions for the 3D model, because the LiDAR can't get the reflecting from glass. We can cover the glass to solve this problem, but we don't want to change the realistic view for Dodge Hall.

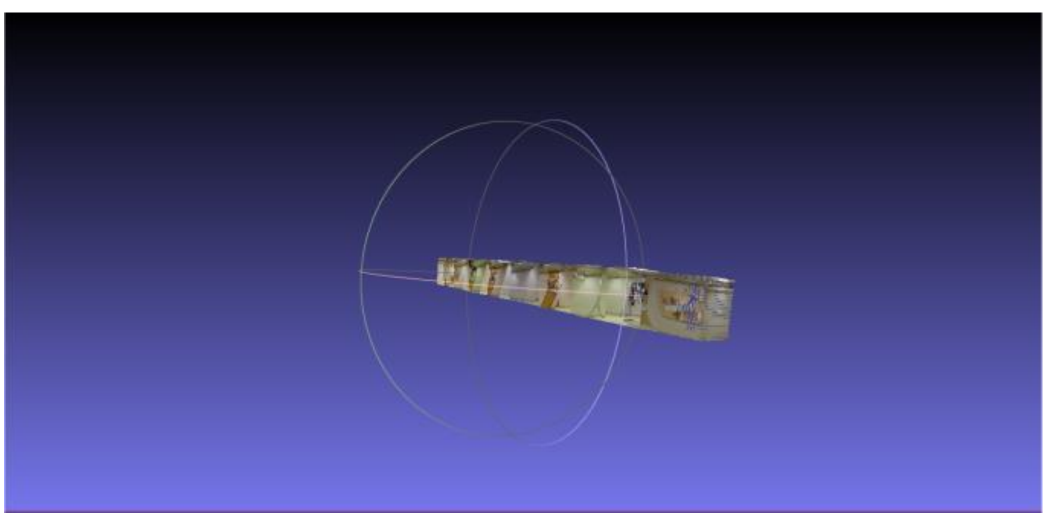


Figure 4.12: One hallway in MeshLab (top).

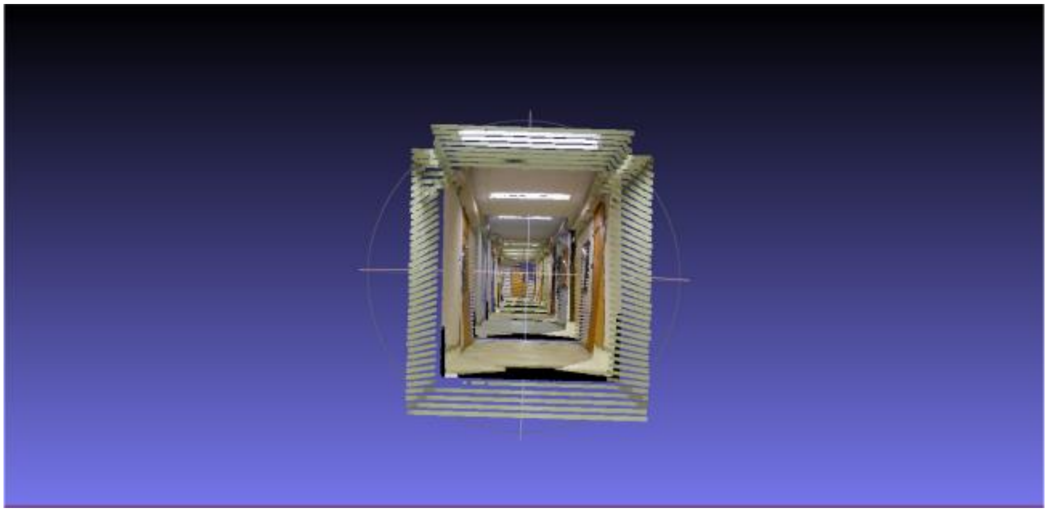


Figure 4.12 (continued): Inside view for the hallway in MeshLab (bottom).

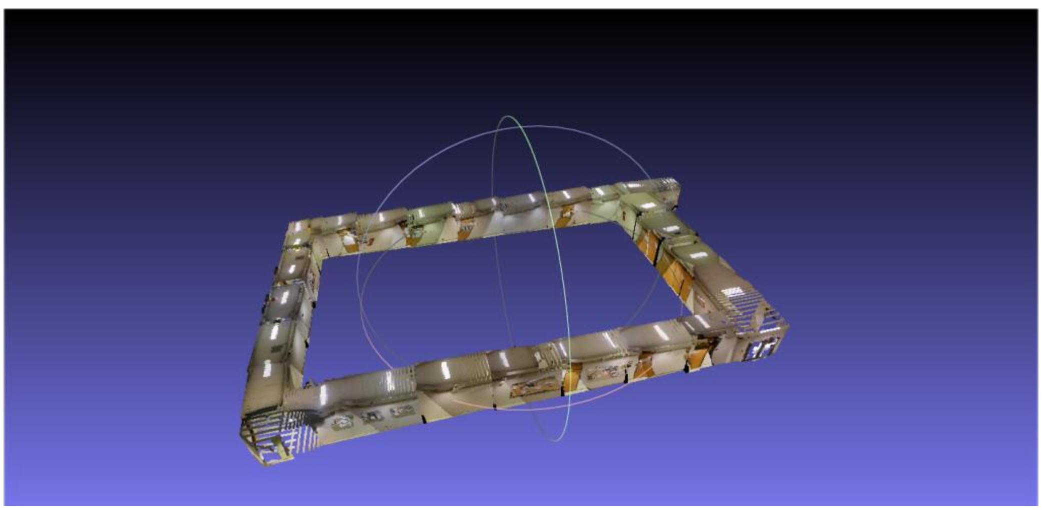


Figure 4.13: 3D model for Dodge Hall in MeshLab.

4.4 Comparisons

4.4.1 SIFT

SIFT (Scale-invariant feature transform) is an algorithm of computer vision to detect and describe local features in images [33]. These features are invariant to image scaling and rotation, and partially invariant to change in illumination and 3D camera viewpoint. It is quite convenient for us to get the features while the images have a little change. We can match different size images and the images that catching from different sensor, such as combine the Lidar intensity data with the camera image. Because the extracted features are belongs to both spatial and frequency domains, they have a little disruption caused by occlusion, clutter, or noise. By using SIFT algorithm, we can get a more accurate result than use other method.

The main steps for applying SIFT algorithm are as follows:

1. Scale-space extreme detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

This method belongs to computer vision and it has many applications nowadays. It can be used in robot localization and mapping, panorama stitching, 3D scene modeling, recognition and tracking. The 3D SIFT-like descriptors can be used for human action recognition and SIFT algorithm can be used to analyze the human brain in 3D magnetic resonance images. The SIFT algorithm was introduced by David Lowe [33].

4.4.2 Comparison with SIFT Method

The SIFT algorithm can extract distinctive features from both the LiDAR intensity image and the camera image. The extracted features of two sensors will be automatically matched in MATLAB. And the color information will register with the LiDAR intensity image. The SIFT algorithm can process the data in a short time. Using SIFT algorithm, the feature extraction and alignment from intensity image and camera image for the same object is quite accurate. The SIFT algorithm is used to extrinsic calibrate the LiDAR data and the camera data by extracting corresponding features automatically. In my research, the rigid transformation matrices are used to extrinsic calibrate the multi-sensor system. We extract rigid-body target from the LiDAR intensity image manually and from the camera image automatically. The SIFT method can easily extract features from the images to find the transformation matrices of both sensors for the same object and finally find the correspondence between the LiDAR data and the camera image. We only use the transformation matrices of the camera in three positions to fuse data. The correspondence can be applied to many situations if the LiDAR and camera are rigidly connected. It has less complexity for my research. The number of calculation step and the data storage is less than using SIFT method. By using SIFT algorithm, there are too much data for each location, which means there are also too many extracted features from the LiDAR intensity image and the camera image. Although using SIFT algorithm can obtain a more accurate result, this method is time-consuming and it needs more storage capacity than my method.

4.4.3 Indoor or Outdoor Mapping

In my thesis, the sensor detect system is designed for indoor mapping. We use the technologies such as LiDAR scanning, photogrammetry and computer vision for indoor mapping. These technologies are used to accurately map Dodge Hall interiors. Indoor mapping has the limitation of the scan range of LiDAR and the limitation of complex obstacles in the indoor environment. Figure 4.14 shows an indoor 3D map reconstruction. The method of that grapy uses depth camera to provide traditional image and depth image for each pixel. It has a natural application to build a full 3D indoor map.

The applications for indoor mapping are limited by the device. When we change to a LiDAR 2D sensor with larger scan range, the fusion of the LiDAR sensor and the camera can be applied in outdoor environment. Figure 4.15 shows an example for

Figure 4.14: Application for indoor mapping.
(http://www.hizook.com/blog/2010/03/28/low-cost-depth-cameras-aka-ranging-cameras-or-rgb-d-cameras-emerge-2010)

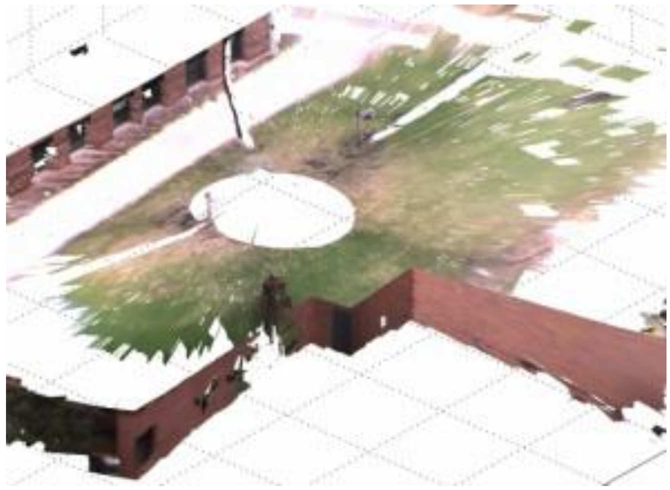


Figure 4.15: Mapping for outdoor environment.
(Mirzaei, Faraz M., Dimitrios G. Kottas, and Stergios I. Roumeliotis. "3D LIDAR–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization." The International Journal of Robotics Research 31.4 (2012): 452-467.)

outdoor mapping. The applications for outdoor mapping include automotive vehicles, architectural mapping, military robots detection, mine exploration and so on.

For indoor mapping, we need a relatively small mobile platform, because the scanning area is not too big for walking and the distance between two adjacent sampling locations are not far away. These assumptions are not valid for automotive vehicles mapping. There are many applications for automotive vehicles to scan and combine the data of the LiDAR sensor and the camera sensor. Ford Company uses this technology on Ford Hybrid to automatically fuse the LiDAR scan data and camera image. The LiDAR mounted on the top of research vehicle can obtain reflections within 200 feet of sensor soundings. Then it can create a three-dimensional map with realistic color from the outdoor environment [23, 29]. The devices for automated vehicle multi-sensor detect system always include a 3D LiDAR sensor like a Velodyne 3D LiDAR scanner and a camera like a Point Grey Ladybug3 omnidirectional camera system. The 3D LiDAR and omnidirectional camera will quickly collect these two types of data while the vehicle is driven around the detecting areas. The SLAM (Simultaneous Localization and Mapping) is applied here. Figure 4.16 shows the devices used for automated vehicle detection [9].

Architectural mapping can apply or the ground or to the air area. We can put the multi-sensor system in front of the building and scan the architecture from all sides of the building to generate a 3D building model. The other way for architectural mapping is using aerial mapping technology. The detecting sensors such as LiDAR sensor, camera and on-board GPS are brought in a plane to receive the reflections, signals and images from the earth's surface. After analyzing the data obtained from detecting sensors, the elevation and geospatial location of the architecture cam be determined. Another

Figure 4.16: The device for outdoor mapping [9].

application is mine mapping robots. It will help people a lot to map the tough environment such as the deep underneath and the surface of the earth. By using a robot for outdoor mapping, we can avoid many dangerous situations such as mine environment, military circumstance and so on.

Comparing indoor mapping with outdoor mapping, the outdoor mapping require a wider field of view from the LiDAR sensor to detect the wall or some objects that far away from the detected system. If we use the LiDAR indoor device for outdoor mapping, the range of the indoor LiDAR cannot reach that far. Then the detected data is the field of view which is within the range of LiDAR indoor sensor. Fusing the data of the multi-sensor system can only construct a 3D map for part of the environment. If we want to use the indoor LiDAR to scan the outdoor environment, the data of many sampling locations must be merged to generate the outdoor map.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusions

The thesis presents a different perspective for fusion data of multi-sensors. When we only use one type of sensor, such as a LiDAR sensor or camera, it provides robust solutions. When we use the data of the LiDAR sensor, it can only provide us a list of 3D points cloud. We can only get the distance information from those points and it is hard to recognize what the object is and what color the object is. When we get the camera image for detecting or mapping, there is not depth information for the object. We can only get the figure or color for the objects, and we cannot get the 3D shape of the objects. It is hard to know the real size for the object and hard to imagine the object in three dimensions. If we use the sensor independently, there will be some difficulties for getting precise results. This thesis demonstrates theoretic and experimental procedures for some common problems like obtaining data, sensor calibration, scan registration, and generation of 3D maps.

Firstly, we obtain the source of the information from different modalities and utilize this information to do some analysis. A manually mobile platform is equipped with two modalities of sensors (a LiDAR sensor and a digital camera) and a control device (servo). We combine the data obtained from two sensors, and we use the rigid-body extrinsic calibration method to enhance the robustness from the two independent data. We have a low-cost, light weight 2D range sensor. This kind of line-scan LiDAR can scan data in a plane. Adding a rolling device allows the line-scan LiDAR to be used as a conventional horizontal scanner with a high angular region of the scan place. After mounting the LiDAR on the top of the servo, the LiDAR sensor provides 3D points for further use. The digital camera can provide panoramas to fuse with the 3D points cloud. To provide a wide front view of the sensor scenes, we set up the camera in three positions to capture images. My contributions here for obtaining the data from two sensors are combining the LiDAR and servo to generate a wider image view. The 2D LiDAR has the advantages of low cost and easy mounting. The function for combining the servo with the LiDAR sensor system is similar to the function of 3D LiDAR. The combined system is more convenient and cheaper than the 3D LiDAR. The view of images obtained from the camera is quite narrow. We generate panoramas by stitching 13 camera images and combining three panoramas in the same LiDAR intensity image. After that, we can generate a 3D points cloud with more color and texture information to match.

The second step is using the extrinsic calibration method to combine the LiDAR sensor with the camera. This allows the 3D points to be projected onto the corresponding camera plane. A checkerboard is placed in front view of the multi-sensor system. We use a rigid-body target, a planar checkerboard pattern for extrinsic calibration of sensor detect system. It requires us to estimate the transformation matrix between the camera and LiDAR scan sensor. After we set up the rigid position for the LiDAR and the camera, we need to keep the relative position for these two sensors. Once we get the transformation matrices, the same calibration matrices can be used for the rest of the locations in the hallway. The method here is to use a fiducial target to get the correspondence between the rigid mounted LiDAR sensor and the camera. Because of the rigid correspondence between the LiDAR and the camera, we can get three rigid transformation matrices. These three transformation matrices can be used for all detecting locations for the hallway when we use the same device in experiments. We use rigid transformation matrices instead of automatically stitching the LiDAR intensity image and panoramas by extracting the features from the data of both sensors. The method we used has less calculated steps.

Thirdly, we use a planar checkerboard pattern to complete the fusion of sensor data associated color and texture information from the camera with 3D points of intensity information from the LiDAR sensor. After we get the correspondence between the two fused sensor data, we need to complete the registration of two sequential scans. Interpolation of fused data is the method to align two sensor data comprised of the co-registered camera and LiDAR data. Then, we can accomplish building the reconstruction of the 3D map of Dodge Hall at Oakland University. The registration requires the knowledge of the correspondence between the spatial coordinates and pixel indices in an image. The pixel is in discrete data format and the spatial coordinates are in grid format. Because the projected LiDAR 2D points are not integers, the 2D points are interpolated to get the nearest integer value for the pixel and then give the pixel value for each LiDAR 2D point.

Finally, we survey Dodge Hall for data collection. Because the LiDAR-servo system can’t get the 3D points while the detect system is moving, we need to set 25 measuring locations in Dodge Hall. In other researches, they record the data from the 3D

LiDAR range sensor and the camera while driving. The advantage of set discrete measuring locations is that we can get more accurate results because the distance between the two recording locations is known.

5.2 Future Work

Future work will include the improvement for the multi-sensor system which means the integration with other sensors. Although 2D LiDAR has the advantages of low cost and light weight, 3D LiDAR sensors like Velodyne 3D LiDAR range sensor can obtain 3D points directly without a servo. We can change the Nikon S3000 digital camera to an omnidirectional camera, because this type of camera has a wider field of view than the common digital camera. If we use the omnidirectional camera in the multi-sensor system, we do not need to generate the panoramas and set up three positions for the camera to separately get the three transformation matrices. It can largely increase the accuracy of the result in the research by using the 3D LiDAR and omnidirectional camera as the combined sensor system [7, 8, 10]. The omnidirectional camera is always used to improve human readability for the 3D map or the 3D model. Our method can be used in extrinsic calibration of the 3D LiDAR and the camera. The more accurate result can be obtained by using more checkerboard patterns or using a checkerboard with more white and black squares. Then, the transformation matrix can accurately transform the camera color information to the LiDAR intensity image.

Fusion of the 3D LiDAR and the camera is one branch of the SLAM algorithm. This algorithm has a wide application for localization and mapping. Simultaneous localization and mapping (SLAM) is a relatively new term, and it can be used in self-driving cars, unmanned aerial vehicles, autonomous underwater vehicles, planetary rovers, newly emerging domestic robots, and even inside the human body. In robotics, the SLAM algorithm is used for the computational problem of constructing or updating a map of an unknown environment. We use laser scanning or visual features to provide details of a great many points within an area from the SLAM sensing method. Rendering SLAM inference is not necessary because by using image registration, the shapes in the points cloud are easily and unambiguously aligned.

REFERENCES

[1] Xiaojin Gong, Ying Lin and Jilin Liu. “3D LIDAR-Camera Extrinsic Calibration Using an Arbitrary Tribedron.” Sensors, 2003. ISSN 1424-8220

[2] Song, Jeong-Heon, et al. "Assessing the possibility of land-cover classification using lidar intensity data." International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences 34.3/B (2002): 259-262.

[3] Park, Yoonsu, et al. "Calibration between color camera and 3D LIDAR instruments with a polygonal planar board." Sensors 14.3 (2014): 5333-5353.

[4] H. Arefi, J. Engels, M. Hahn and H. Mayer. “Levels of Detail in 3D Building Reconstruction from LiDAR Data.” International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences Vol. XXXVII. (2008) Part B3b

[5] Scaramuzza, Davide, Ahad Harati, and Roland Siegwart. "Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes."Intelligent Robots and Systems, IROS.IEEE/RSJ International Conference on. IEEE, 2007.

[6] Sheh, Raymond, et al. "A low-cost, compact, lightweight 3d range sensor."Australian Conference on Robotics and Automation. 2006.

[7] Pandey, Gaurav. “An Information Theoretic Framework for Camera and Lidar Sensor Data Fusion and its Applications in Autonomous Navigation of Vehicles.” Diss. The University of Michigan, 2014.

[8] Pandey, Gaurav, et al. "Extrinsic calibration of a 3d laser scanner and an omnidirectional camera." 7th IFAC Symposium on Intelligent Autonomous Vehicles. 2010.Vol. 7. No. 1.

[9] Pandey, Gaurav, James R. McBride, and Ryan M. Eustice. "Ford campus vision and lidar data set." The International Journal of Robotics Research 30.13 (2011): 1543-1552.

[10] Mirzaei, Faraz M., Dimitrios G. Kottas, and Stergios I. Roumeliotis. "3D LIDAR–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization." The International Journal of Robotics Research 31.4 (2012): 452-467.

[11] Gong, Xiaojin, Ying Lin, and Jilin Liu. "Extrinsic calibration of a 3D LIDAR and a camera using a trihedron." Optics and Lasers in Engineering 51.4 (2013): 394-401.

[12] García-Moreno, Angel-Iván, et al. "LIDAR and panoramic camera extrinsic calibration approach using a pattern plane." Pattern Recognition. Springer Berlin Heidelberg, 2013. 104-113.

[13] Kuder, Marko, Marjan Šterk, and Borut Žalik. "Point-based rendering optimization with textured meshes for fast LiDAR visualization." Computers & Geosciences 59 (2013): 181-190.

[14] Mastin, Andrew, Jeremy Kepner, and J. Fisher. "Automatic registration of LIDAR and optical images of urban scenes." Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009.

[15] Pluim, Josien PW, JB Antoine Maintz, and Max A. Viergever. "Image registration by maximization of combined mutual information and gradient information." Medical Image Computing and Computer-Assisted Intervention–MICCAI 2000. Springer Berlin Heidelberg, 2000.

[16] Mastin, Dana Andrew. "Statistical methods for 2D-3D registration of optical and LIDAR images." Diss. Massachusetts Institute of Technology, 2009.

[17] Torr, Philip HS, and Andrew Zisserman. "MLESAC: A new robust estimator with application to estimating image geometry." Computer Vision and Image Understanding 78.1 (2000): 138-156.

[18] Hartley, Richard, and Andrew Zisserman. "Multiple view geometry in computer vision." Cambridge university press, 2003.

[19] Badamikar, Neeraj S. "Automatic Registration of Multiple Texel Images to Form a 3-Dimensional Texel Image." (2014).

[20] Huitl, Robert, et al. "TUMindoor: An extensive image and point cloud dataset for visual indoor localization and mapping." Image Processing (ICIP), 2012 19th IEEE International Conference on. IEEE, 2012.

[21] Alismail, Hatem, L. Douglas Baker, and Brett Browning. "Automatic calibration of a range sensor and camera system." 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on. IEEE, 2012.

[22] Zhou, Lipu, and Zhidong Deng. "Extrinsic calibration of a camera and a lidar based on decoupling the rotation from the translation." Intelligent Vehicles Symposium (IV), 2012 IEEE. IEEE, 2012.

[23] Florez, Sergio Alberto Rodriguez, et al. "Multi-modal object detection and localization for high integrity driving assistance." Machine vision and applications 25.3 (2014): 583-598.

[24] Zhou, Lipu. "A new minimal solution for the extrinsic calibration of a 2D LIDAR and a camera using three plane-line correspondences." Sensors Journal, IEEE14.2 (2014): 442-454.

[25] Budge, Scott E., Neeraj S. Badamikar, and Xuan Xie. "Automatic registration of fused lidar/digital imagery (texel images) for three-dimensional image creation." Optical Engineering 54.3 (2015): 031105-031105.

[26] Debattisti, Stefano, Luca Mazzei, and Matteo Panciroli. "Automated extrinsic laser and camera inter-calibration using triangular targets." Intelligent Vehicles Symposium (IV), 2013 IEEE. IEEE, 2013.

[27] Zhou, Lipu, and Zhidong Deng. "A new algorithm for the extrinsic calibration of a 2D LIDAR and a camera." Measurement Science and Technology 25.6 (2014): 065107.

[28] Miller, Stephen, Alex Teichman, and Sebastian Thrun. "Unsupervised extrinsic calibration of depth sensors in dynamic scenes." Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. IEEE, 2013.

[29] Rodriguez, F., Vincent Fremont, and Philippe Bonnifait. "Extrinsic calibration between a multi-layer LIDAR and a camera." Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on. IEEE, 2008.

[30] Mirzaei, Faraz M., Dimitrios G. Kottas, and Stergios I. Roumeliotis. "3D LIDAR–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization." The International Journal of Robotics Research 31.4 (2012): 452-467.

[31] Liu, Timothy, et al. "Indoor localization and visualization using a human-operated backpack system." Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on. IEEE, 2010.

[32] Bodensteiner, Christoph, et al. "Monocular camera trajectory optimization using LiDAR data." Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on. IEEE, 2011.

[33] David G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.