# Lidar-Camera Calibration for Self Driving Cars

Student Name: Aarushi Agarwal
Roll Number: 2016216

BTP report submitted in partial fulfillment of the requirements
for the Degree of B.Tech. in Computer Science & Engineering

on May 30, 2020

**BTP Track**: Research

**BTP Advisor**
Dr. Saket Anand

Indraprastha Institute of Information Technology
New Delhi

## Students' Declaration

I hereby declare that the work presented in the report entitled **Lidar-Camera Calibration for Self Driving Cars** submitted by me for the partial fulfillment of the requirements for the degree of *Bachelor of Technology* in *Computer Science & Engineering* at Indraprastha Institute of Information Technology, Delhi, is an authentic record of work we carried out under guidance of **Dr. Saket Anand**. Due acknowledgements have been given in the report to all material used. This work has not been submitted anywhere else for the reward of any other degree.

**..................................** **Place & Date: ............................**
**Aarushi Agarwal**

## Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

**.............................** **Place & Date: ............................**
**Dr. Saket Anand**

**Abstract**

Now-a-days, many applications require the reconstruction of 3D structure of a scene captured by sensors like Camera. It involves linking the world coordinate frame with the corresponding projection in the image plane and solving for the camera's parameters. Such a process is referred to as geometric camera calibration. We shall explore various techniques for performing the camera calibration.

Further in Autonomous Systems, sensors like cameras and LiDARs are used to capture different types of information in the world. We adopt an approach to place a calibration pattern which lies in the field of view of LiDAR and Camera simultaneously. To do the fusion of the data extracted from both, we need to perform cross calibration between them, i.e bring the coordinate frames into a common frame of coordinate system. Hence, we calculate the transformation vectors between LiDAR and Camera and perform a series of experiments to validate our results. In this paper, we shall try to survey the basic concepts and implement various strategies for Camera Calibration and Cross Calibration between LiDAR and Camera. We also did a literature study over different domains like Cross Calibration, Auto Calibration and Sensor Fusion to get an idea of the possible future work.

Keywords: LiDAR, Camera, Calibration, Cross Calibration

# Acknowledgments

I am deeply grateful to my advisor Dr. Saket Anand for giving me an opportunity to work with him and introducing me to the problem. He has been guiding me constantly with his invaluable suggestions on how to approach the research problem.

# Contents

# Chapter 1

# Geometric Camera Calibration

Now-a-days, many applications require the reconstruction of 3D structure of a scene captured by sensors like the Camera. Calibration can be used for the same. Camera Calibration is referred to as a procedure of linking the world coordinate frame with the corresponding projection in the image plane and solving for the camera's parameters. It is performed on an object whose geometry in the 3D world is known to us. One such pattern which is used here is the Checkerboard Plane [6] wherein the mapping is generated between the vertices from the world frame to the image plane by looking at the intersection of the horizontal and vertical lines in the image.

Geometric camera calibration is composed of two parts :-

- Going from the known world coordinate frame to camera's 3D coordinate frame - **Extrinsic Parameters**
- From the 3D coordinate frame of the Camera to the 2D Image Plane - **Intrinsic Parameters**

## 1.1 Extrinsic Parameters

This tells us where the camera is located in the world and about its pose. The transformation from the world coordinate system to camera coordinate system is referred to as $T_W^C$. The transformation matrix can be represented as :-

$$T_W^C = [R_W^C t_W^C]$$

where :

$$R_W^C \; is \; a \; 3 \times 3 \; orthogonal \; rotation \; matrix$$

$$t_W^C \; is \; a \; 3 \times 1 \; translation \; vector$$

The coordinates of the point in the world frame of reference is represented as $[X_W \; Y_W \; Z_W]$ and in the 3D coordinate frame of reference as $[X_C \; Y_C \; Z_C]$. The relation can be seen as follows :

$$[X_C \; Y_C \; Z_C] = R_W^C [X_W \; Y_W \; Z_W] \; + \; t_W^C$$

Hence, the Extrinsic parameters can be represented as :

$$T \; = \; [\; R \mid t \;]$$

$$T = \begin{bmatrix} r11 & r12 & r13 & t1 \\ r21 & r22 & r23 & t2 \\ r31 & r32 & r33 & t3 \end{bmatrix}$$

## 1.2 Intrinsic Parameters

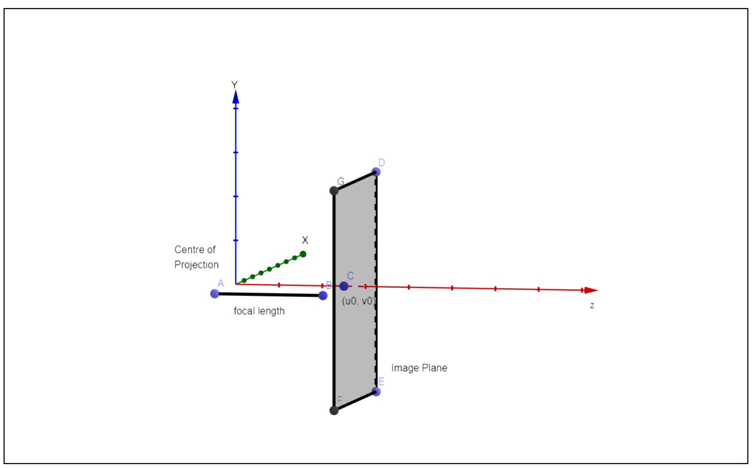


Figure 1.1: Camera coordinate frame

These are the internal properties of the camera and are denoted by the matrix K where :

$$K = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Here $f_x$ and $f_y$ denote the focal length of the camera and $u_0$ and $v_0$ be the coordinates of the optical centre of the camera. In the diagram we can see that the distance between the centre of projection and the centre of the image plane is termed as the focal length.

# Chapter 2

# Zhang's Method Of Calibration

The transformation equation from 3D plane to 2D plane (Homogeneous coordinates) can be written as :

$$P_i = K[R\ T]P_w$$

where :

$$P_i\ :\ point\ in\ image\ frame$$

$$P_w\ :\ point\ in\ world\ frame$$

Here, K [R T] denotes the camera projective matrix.

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\begin{bmatrix} r11 & r12 & r13 & t1 \\ r21 & r22 & r23 & t2 \\ r31 & r32 & r33 & t3 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

In checkerboard, we know that the z coordinate of the checkerboard is 0. Hence we can say that Z = 0. Hence, we can write the above equation as :-

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\begin{bmatrix} r11 & r12 & t1 \\ r21 & r22 & t2 \\ r31 & r32 & t3 \end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$Considering\ M = K\begin{bmatrix} r11 & r12 & t1 \\ r21 & r22 & t2 \\ r31 & r32 & t3 \end{bmatrix}\ and\ M = [m1\ m2\ m3\ ],\ we\ can\ say\ that$$

$$[m1\ m2\ m3] = K[r1\ r2\ t]$$

Hence,

$$r1 \; = \; K^{-1} \, m1$$

$$r2 \; = \; K^{-1} \, m2$$

Since Rotation matrix is an orthogonal matrix, $r1^T r2 = 0$ and $|r1| = |r2| = 1$

$$m1^T K^{-T} K^{-1} m2 = 0$$

$$m1^T K^{-T} K^{-1} m1 \; - \; m2^T K^{-T} K^{-1} m2 = 0$$

Substitute $B = K^{-T}K^{-1}$ (Use Cholesky Decomposition to compute B) which is a positive definite and symmetric matrix . Hence we need the values of upper triangle in B. Let :

$$B = \begin{bmatrix} b11 & b12 & b13 \\ b21 & b22 & b23 \\ b31 & b32 & b33 \end{bmatrix}$$

Define b = (b11 , b12, b13 , b22, b23 , b33) . Hence, we can combine the above equations to a form of Vb = 0, where :

$$V = \begin{bmatrix} v12^T \\ v11^T - v22^T \end{bmatrix}$$

where,

$$vij^T = [m_{i1}m_{j1} \; m_{i1}m_{j2} + m_{i2}m_{j1} \; m_{i1}m_{j3} + m_{i3}m_{j1} \; .......]$$

We find that the extrinsic and the intrinsic parameters of the Camera can also be found using the method "calibrateCamera" in the library OpenCV in python. This method requires data from different positions of the calibration pattern and a mapping of the world coordinates and the corresponding image coordinates obtained from the camera.

We take the checkerboard plane as the calibration pattern which is a 2D plane. Hence the points taken in the world frame coordinates are (0,0,0) , (0,1,0) , (0,2,0)……(1,2,0) ,(1,3,0).....(2,0,0), (2,1,0)...etc .

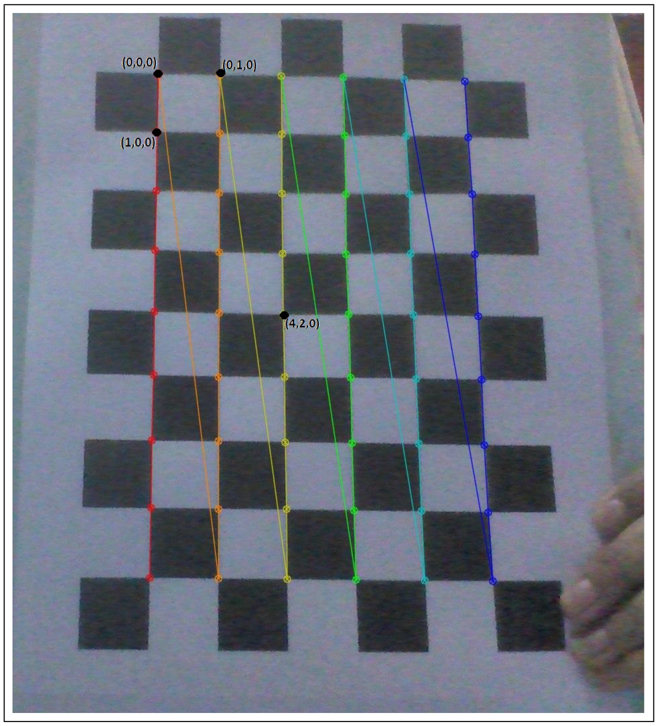


Figure 2.1: Checkerboard pattern with coordinates of vertices in world frame

**NOTE** : The z coordinate of all the points will be 0 , since we assume the pattern to be a x-y plane. The corresponding image points are obtained from the openCV method findChessboard-Corners.

# Chapter 3

# Cross Calibration between LiDAR and Camera

In systems like Autonomous Vehicles, various sensors like cameras and LiDARs are used to capture different types of information in the world. While the camera helps to cover the 2D element of a scene, like the appearance, color etc , the LiDAR helps in understanding the structure of the pattern.

In our experiment with the checkerboard, both LiDAR and camera are looking at the calibration pattern in their respective frame of references. To do the fusion of the data extracted from both, we need to perform cross calibration between them, i.e bring the coordinate frames into a common frame of coordinate system.

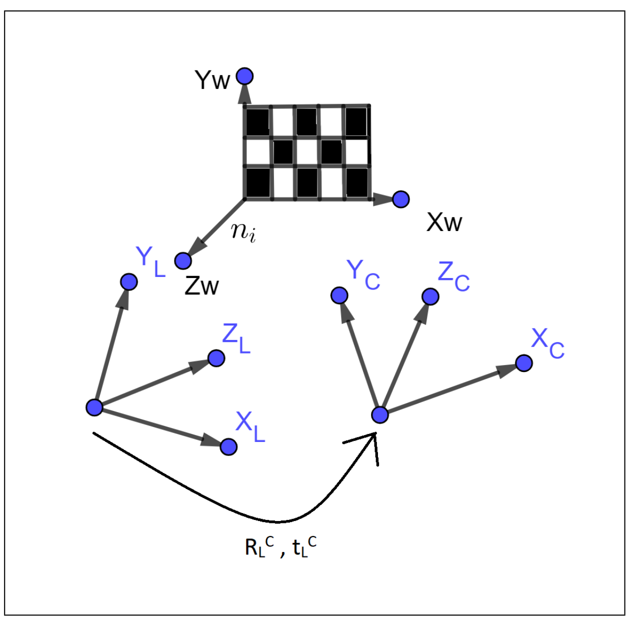


Figure 3.1: Calibration Pattern from different coordinate frames

## 3.1 Estimating Rotation vector

$R_L^C$ is Rotation vector from LiDAR frame to Camera frame. Now, to perform the cross calibration, we observe that we need a mapping of points on the calibration pattern in both the coordinate frames. However, in the case of LiDAR, the data is in the form of 3D points lying on the pattern in the form of horizontal lines. From this data, we cannot tell which point is the vertex of the checkerboard plane. Hence, we need to find some mechanism which can help us build correspondence between both the frames.

We observe that we can find the normal vector of the 2D Checkerboard plane in both the coordinate frames. Although it would be the same physical vector, the representation in the LiDAR coordinate frame ($n_i^L$) and in the Camera coordinate frame ($n_i^C$) will be different. The relation can be define is as follows :-

$$R_L^C A \; = \; B \tag{3.1}$$

$$A = [n_1^L, n_2^L, n_3^L \dots .n_m^L]$$

$$B = [n_1^C, n_2^C, n_3^C \dots .n_m^C]$$

Where A and B are the normal vectors of the various calibration patterns in the Lidar and Camera frame of reference. (m = number of views of the calibration pattern)

**<u>NOTE</u>** : Since the normal vector is just a direction and has nothing to do with the position, we can only define the Rotation transformation in this way.

### 3.1.1 Implementing equation (3.1)

To implement this method, we first need to find $n_i^L$ and $n_i^C$. From the above parts, we observed that we have obtained the translation and the rotation matrices of the world to camera frame of reference through Camera Calibration.

Considering the fact that our calibration pattern is a 2D plane (Assumption : x-y plane) , the normal vector in world frame of reference would be :

$$n_i^W \; = \; [0\;0\;1]$$

The corresponding normal ($n_i^C$) in Camera frame of reference will be :

$$n_i^C = R_W^C n_i^W \tag{3.2}$$

To calculate the corresponding normal ($n_i^L$) in the LiDAR frame of reference, we find the equation of the plane from the point data of LiDAR. If the equation of plane is of the form

$$aX \ + \ bY \ + \ cZ \ = \ d$$

where,

$$X = [x_1, x_2......x_n] \ \ (set \ of \ x \ coordinates \ of \ all \ the \ points \ in \ i_{th} \ image)$$
$$Y = [y_1, y_2......y_n] \ \ (set \ of \ y \ coordinates \ of \ all \ the \ points \ in \ i_{th} \ image)$$
$$Z = [z_1, z_2......z_n] \ \ (set \ of \ z \ coordinates \ of \ all \ the \ points \ in \ i_{th} \ image)$$

then by knowledge of 3D Geometry, the normal vector can be found as :- [a b c]. To find the solution, we can view the above equation as follows :

$$aX + bY + cZ = d$$

or

$$aX + bY - d = -cZ$$

Putting c = -1 ,

$$aX + bY - d = Z$$
$$[X \ Y \ \ -1]^T [a \ b \ d] = Z \tag{3.3}$$

This equation can be seen as a system of linear equation, Ax = b , where A = $[\mathrm{X\ Y\ -1}]^{\mathrm{T}}$ and x = [a b d] and b = Z. To solve the linear equation , we use the least squares solution method.

$$Ax = b$$
$$(A^T A)x = A^T b$$
$$(A^T A)^{-1}(A^T A)x = (A^T A)^{-1} A^T b$$
$$x = (A^T A)^{-1} A^T b$$

Hence we find the value of a, b, d from the above method. The equation of normal would be [a b c] or [a b -1] (Since we took c = -1). After getting the normal vector , we normalize it to get the unit vector $n_i^L$ in the LiDAR coordinate system.

**NOTE** : By putting c = -1 , we fixed the direction of the normal. The calculated normal can be either in the direction of the plane or opposite to it. Hence, if

$$\lambda = n.p < 0$$

where n is the calculated normal vector and p is any point lying on the calibration plane.

This means that the direction of the plane from origin and the calculated normal vector are in opposite directions. Hence the actual normal vector will be -n .

We can now apply the equation 3.1 to get the transformation Rotation vector $R_L^C$. Solving equation 3.1 using SVD Method [7] [8] :-

1. Compute the $\mathrm{Centroid_A}$ and $\mathrm{Centroid_B}$ where
   (a) $Centroid_A = \frac{1}{N}\Sigma_{i=1}^{N} A^i \; where \; A^i = n_i^L$
   (b) $Centroid_B = \frac{1}{N}\Sigma_{i=1}^{N} B^i \; where \; B^i = n_i^C$
2. Let H = (A - $\mathrm{Centroid_A}$) (B - $\mathrm{Centroid_B}$)$^{\mathrm{T}}$
3. Compute $U \sum V^T = H$
4. Define $R_L^C = VU^T$

## 3.2 Estimating Translation vector

$t_L^C$ is Translation vector from LiDAR frame to Camera frame.

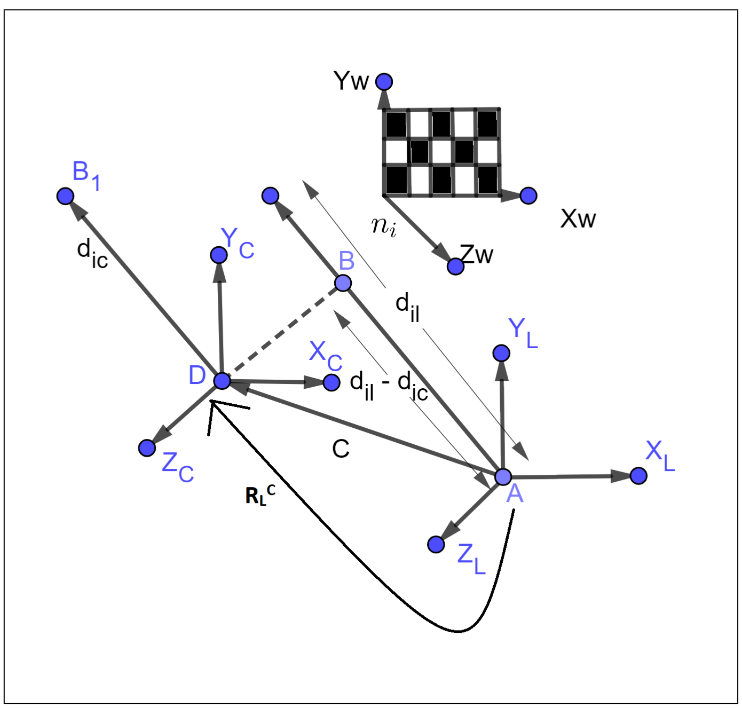


Figure 3.2: Finding Translation Vector

For finding the $t_L^C$, we take into account :

1. $d_i^L$ : The distance of LiDAR from the checkerboard along the normal of the pattern ($n_i^L$).
2. $d_i^C$: The distance of Camera from the checkerboard along the normal of the pattern ($n_i^C$).

Now, we know that the $n_i^L$ and $n_i^C$ are the same physical vector ($n_i$ of the checkerboard) but in different coordinate frames. Hence visualize $d_i^L$ - $d_i^C$ as the distance between Lidar and Camera along the normal of the Calibration Pattern.

If we denote the centre of Camera in frame LiDAR coordinate system as C, we can say that $C.n_i^L$ is the distance of Camera from Lidar along the normal of the pattern.

Hence, we can show that :

$$C.n_i^L = d_i^L - d_i^C$$
$$(n_i^L)^T C = d_i^L - d_i^C \tag{3.4}$$

Now we know that the point C is a vector in the LiDAR frame of reference. In the camera's frame of reference , it corresponds to [0,0,0].

Hence while transforming from LiDAR to Camera, we can say that :

$$R_L^C C + t_L^C C = [0,0,0]$$
$$t_L^C = -R_L^C C \tag{3.5}$$

### 3.2.1 Implementing equation (3.4) and (3.5)

1. In order to calculate $d_i^C$, we use that the translation vector $t_W^C$ which denotes the translation of a vector from world coordinate frame to the Camera frame. Hence the Euclidean distance of the plane is defined as : $\sqrt{x2 + y2 + z2}$ where $t_W^C = [x\ y\ z]$.
2. To calculate $d_i^L$, we know that in the equation in the equation $aX + bY + cZ = d$ , d denotes the distance of the plane from the origin ([a b c] is a unit vector). We have already found the equation of the plane by solving equation (3.3). Hence, we can get $d_i^L$ .

Now, we can solve equation (3.4) and (3.5) to get $t_L^C$.

# Chapter 4

# Verification of Results

After getting the Transformation matrix , $T = [R_L^C \; t_L^C]$, we need to evaluate our performance.

## 4.1 Evaluating Rotation Vector $(R_L^C)$

1. Orthogonal Matrix :- Rotation matrix must be an orthogonal matrix and hence, it should follow the rule $(R_L^C) \times (R_L^C)^T = I$(identity matrix) . The product comes out to be 0.9999999999999998 .

2. Visualization :- By visualizing the alignment between the normal vector in camera frame $(n_i^C)$ and the rotated vector from LiDAR frame of reference $(R_L^C n_i^L)$ , we get the following results :-

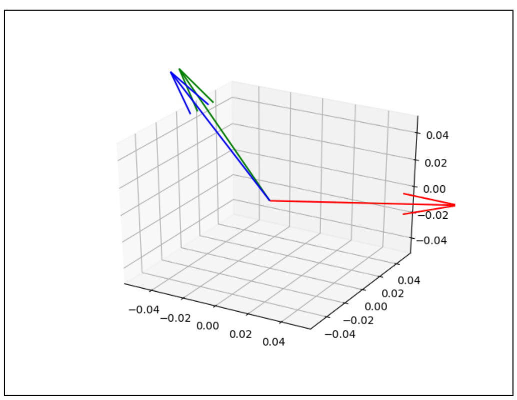


Figure 4.1: Normal Vectors in different frames

**Legend**

Red - Normal in LiDAR frame of reference

Blue - Normal in Camera frame of reference

Green - Rotated vector from LiDAR to camera frame

3. Cosine distance :- The cosine distance between two vectors X and Y can be defined as : $1 - X.Y/(XY)$ This is equal to $1 - cos\theta$ where $\theta$ is the angle between the two vectors.

   We tried to compute the cosine distance between $R_L^C n_i^L$ and $n_i^C$. Cosine distances for 10 images are as follows :

   [0.0008944587487264011, 0.0007953579465221905, 0.0008141069875786666, 0.005811864510685161, 0.003002343073866709, 0.002690492184996396, 0.015968772795163177, 0.002913338053310066, 0.004877972195971636, 0.0018296812187843514 ]

   As we can see that the distances are near to 0, which shows that the vectors are very close to each other.

## 4.2 Evaluating Translation Vector $(t_L^C)$

To verify the translation vector, we randomly sample a few images (15) from all the images (29) and try to calculate the translation vector for the sampled population. We repeat this process a number of times (1000). Finally we get a set of translation vectors for all the samples.

$$sampled_{translationLtoC} = [t1_L^C, t2_L^C, t3_L^C, t4_L^C, t5_L^C .................. tn_L^C]$$

Here n = 1000 (number of times sampling is performed)

We try to compute the standard deviation (variation) between all the translation vectors. The standard deviation is around :-

array([3.32613569, 5.01060966, 1.80145976])

We find that the original $t_L^C$ lies within the range $[(\mu - \sigma), (\mu + \sigma)]$ where

$$\mu : mean\ of\ sampled_{translationLtoC}$$

$$\sigma : standard\ deviation\ of\ sampled_{translationLtoC}$$

Also, the difference between the vector and the $t_L^C$ vector came out to be :-

array([ 0.09731686, -0.24683577, 0.22417081])

As can be seen that the translation vector is quite close to the mean vector. Hence, we can validate our results.

# Chapter 5

# Literature Survey

## 5.1 Cross Calibration

### 5.1.1 Online Cross-Calibration of Camera and LIDAR (ICCP - 2017) [2]

In AV Systems, drift can affect the sensor's position. In this paper, a method is developed which constantly tracks the two sensors and adjust their value of cross calibration. This process is known as Online Calibration. It involves overlapping of edges from LiDAR data over the edges in images. The system is unaffected by the speed of vehicle.

The drawback of existing methods is that they use specific calibration targets which aren't available if we want to re-calibrate (when the sensors undergo a drift). Also, re-calibration is a time consuming process. This paper uses edges as features for finding correspondence between LiDAR and Camera data. They propose edge detection algorithms for both images and range image. They also performs online calibration by applying a motion correction algorithm - assumes that the LiDAR point cloud is not taken at same time. They record different timestamp for each point and adjusts the position of point accordingly. Hence this method of online calibration helps detecting mis calibrations with sensor drift and able to self adjust the parameters.

## 5.2 Sensor Fusion

### 5.2.1 Multi-View 3D Object Detection Network for Autonomous Driving (CVPR - 2017) [3]

In this paper, a MultiView 3D networks (MV3D) has been proposed that takes LiDAR point clouds and camera images as input and predicts the 3D bounding boxes. The network is composed of two subnetworks :-

1. 3D proposal network - generates highly accurate 3D candidate boxes
2. Multi view feature fusion - Extracts features region wise

The Network performs 3D box regression which helps in predicting accurate 3D location , size and orientation. Apart from it it performs Object Detection of the various 3D objects. The performance has been evaluated on the KITTI [4] dataset and is said to outperform the existing methods.

### 5.2.2 Online Camera LiDAR Fusion and Object Detection on Hybrid Data for Autonomous Driving (IVS - 2018) [1]

Fusion involves combining various sensors data to get a single representation of the environment. Each sensor has different uses - RGB cameras perceive color and texture, LiDAR provide distance information, RADARs provide velocity and distance and work well in adverse weather conditions, but have low resolution. In this paper, the fused data is used for object detection. There are two types of known fusion techniques :-

1. HLF (High level fusion) - detects objects with each sensor separately and combines the detection results , discards classifications with low confidence values.
2. LLF (Low level fusion) - combines low level data from different sensor types and then perform fusion.

They project the LiDAR point cloud onto RGB image , upsample it and extract features. Further, Faster RCNN is modified for improved object detection and classification An outline is provided on how Radar can be added to the fusion pipeline. The method provides better results on KITTI [4] dataset and an improvement over the existing calibration method of Levinson and Thrun [5].

# Chapter 6

# Conclusion

Hence, we find that we were able to have an in-depth understanding of Camera Calibration. We studied various methods to calculate the intrinsic and extrinsic parameters and tried to learn the Zhang method. We further went on to perform Cross Calibration between Camera and LiDAR. We got a very good understanding after visualizing the normals of the calibration pattern in LiDAR and Camera frame of reference and computing the transformations. We were able to successfully calculate the Rotation and the Translation transformation vector between the two sensors. After getting the vectors we also went on to verify our computation by using different methods. We found that our calculations were quite close to the expected results and hence validating the outputs. Further, the literature study helped us to discover and get a good overview of the existing methodologies in the respective domains.

# Bibliography


[1] Banerjee, K., Notz, D., Windelen, J., Gavarraju, S., and He, M. Online camera lidar fusion and object detection on hybrid data for autonomous driving. pp. 1632–1638.

[2] Blaga, B.-C.-Z., and Nedevschi, S. Online cross-calibration of camera and lidar. *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)* (2017), 295–301.

[3] Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. Multi-view 3d object detection network for autonomous driving, 2016.

[4] Geiger, A. Are we ready for autonomous driving? the kitti vision benchmark suite. pp. 3354–3361.

[5] Levinson, J., and Thrun, S. Automatic online calibration of cameras and lasers.

[6] Zhang, Z. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 11 (2000), 1330–1334.

[7] https://igl.ethz.ch/projects/ARAP/svd_rot.pdf

[8] https://www.ltu.se/cms_fs/1.51590!/svd-fitting.pdf