

Bachelor's Thesis

LiDAR Extrinsic Calibration using a Cubic Target

School of Mechanical and Control Engineering
Handong Global University

Hyeong-Seok Song

Sun-Been Moon

LiDAR Extrinsic Calibration using a Cubic Target

A Bachelor's Thesis

Submitted to the School of
Mechanical and Control Engineering of
Handong Global University

Hyeong-Seok song

Sun-Been Moon

December 2020

This certifies that the bachelor's thesis is approved.

Thesis Advisor: Ph.D. Yeong-Keun Kim

The Dean of Faculty: Ph.D. Kwon-Yeong Lee

School of Mechanical and Control Engineering

Handong Global University

December 2020

CONTENTS

Extended Abstract	i
1. 서 론	2
1.1 연구목표	2
1.2 연구배경	2
1.3 연구요약	3
2. 연구 방법	5
2.1 LiDAR 포인트 클라우드 상의 타겟 특징점 추정	5
2.2 Reference와 타겟 간의 상대위치 추정	오류! 책갈피가 정의되어 있지 않습니다.
3. 연구 결과	11
3.1 시뮬레이션 결과	11
3.2 실험 결과	13
4. 토 의	15
4.1 시뮬레이션 결과 분석	15
4.2 실험 결과 분석	16
5. 결 론	17
6. References	18

Abstract

This paper presents simulation analysis for the LiDAR extrinsic calibration method using a cubic target object. The proposed algorithm can estimate the alignment pose of the LiDAR body from the target object and the vehicle body. Simulations were conducted to analyze the accuracy and precision of the proposed algorithm. Results showed that the proposed algorithm can estimate the LiDAR pose with a precision less than 0.1 degrees and 2mm.

1. 서 론

1.1 연구목표

본 연구의 목적은 기존 캘리브레이션 연구와는 다르게 3차원 큐브를 타겟으로 사용하며, LiDAR 단일 센서를 이용한 캘리브레이션 알고리즘을 구현하는 것이다. 시뮬레이션과 실험을 통해 상대 위치 추정 정확도를 분석함으로써 알고리즘의 유효성을 검증하고자 한다. 본 연구의 알고리즘은 자동화율을 최대화하며, 센서 간의 융합을 넘어 LiDAR 단일 센서만의 역량으로 캘리브레이션의 정확도를 높이하고자 한다. 캘리브레이션 알고리즘은 추후 ADAS 센서들이 정확한 위치에 부착되어 있는 지 검사하는 생산라인 속 장비에 활용될 전망으로 보고 있다.

1.2 연구배경

국내외 다수의 자동차 산업에서 주행보조 및 자율주행시스템에 대한 관심이 높아지고 있다. ADAS 장착이 의무화되고 있으며, 그에 따른 기술개발이 필수적이다. LiDAR 센서는 ADAS의 핵심 부품으로, mm의 작은 부착오차도 먼 거리에서는 큰 거리 오차가 발생시킨다. 이에 따라, 생산라인 상에서 LiDAR 센서의 정확한 부착 테스트는 필수적인 작업이다. LiDAR 단일 센서의 상대위치를 Ground Truth와 센서 간의 변환 행렬을 구하는 캘리브레이션 연구가 진행되고 있다. 본 연구에서는 LiDAR(Light Detection And Ranging) 센서의 보다 정확하고, 자동화율이 높은 캘리브레이션 연구를 진행할 것이다.

기존의 LiDAR 캘리브레이션 연구는 intrinsic parameter 캘리브레이션과 extrinsic parameter 캘리브레이션으로 나뉜다. Intrinsic parameter 캘리브레이션은 라이다 내부 parameter를 조정한다.[1] 하지만, 본 연구의 목적은 LiDAR의 extrinsic parameter를 조정하는 데 있다.

기존의 LiDAR 캘리브레이션 연구는 주로, LiDAR-카메라 두 센서 융합으로 진행되었다. 위 캘리브레이션 연구는 타겟의 모양에 따라 나뉘어진다. 사각형 평면 보드를 이용하거나[2, 3], 체커 평면 보드를 이용한 연구들이 있다[4, 5]. 그러나, 사각형 평면의 보드를 이용할 경우, LiDAR의 데이터로부터 타겟을 특징점, 모서리 검출이 어렵다는 문제가 있다. 특히, 산업용으로 이용되는 LiDAR의 경우 낮은 resolution으로 인해, vertical의 채널의 수가 적기 때문에 모서리 검출이 더 어렵다. 체커 보드의 경우에는 검은색과

흰색의 패턴 반복으로 인해 노이즈가 다수 발생한다는 문제점이 있다. 노이즈로 인해 체커 보드의 특징점을 잡기 어렵다. 기존의 캘리브레이션 연구를 몇 가지 소개하자면 아래와 같다.

M.Velas *et al*의 연구[3]는 원, 사각형, 삼각형으로 다양한 기하학적 모양으로 구멍이 뚫린 평면 보드를 타겟으로 사용하였다. 연구에서 제안된 알고리즘은 RANSAC을 적용하여 LiDAR 포인트 클라우드 데이터에서 모서리를 검출하고, 카메라에서는 Hough transform을 이용한 3D marker detection을 이용한다. 찾아낸, 특징들을 cross correlation을 이용하여 상대위치를 찾아 내었다.

Geiger, A. *et al* 연구[4]는 체커 보드를 여러 곳에 부착하여, 단일 타겟이 아닌 여러 개의 체커 보드를 타겟으로 이용하였다. 연구에서 제안된 알고리즘은 체커 보드의 seed point를 먼저 찾고, iteration을 이용하여 energy function의 높은 gradient 값을 찾아 전체 체커보드를 파악한다. Global registration을 통해 단일 타겟에서 여러 개의 타겟을 찾는다.

최근에, 3차원 큐브형 타겟물을 이용하며 기존 planar calibration보다 정밀한 결과를 산출한 연구가 발표되었다.[6] 이에 따라 본 연구에서는 3차원 큐브형 타겟을 이용한 알고리즘을 구현하고 시뮬레이션과 실험을 통해 검증할 것이다. 3차원 큐브형 타겟을 이용하여, 저해상도의 산업용 LiDAR 캘리브레이션의 단점을 극복하고자 한다.

또한, 기존의 카메라-LiDAR 센서 융합을 통한 캘리브레이션 연구들이 진행되었으나, LiDAR 단일 센서만을 이용한 캘리브레이션 연구가 발표되었다.[7] 여러가지 센서를 융합할 수 없는 상황에서도, 단일 센서를 이용한 캘리브레이션 알고리즘 개발되었다. 그 연구에서는 낮은 FOV를 극복하기 위해, PD sensor를 이용하였다. 따라서, 본 연구에서는 센서간 융합이 아닌, LiDAR 단일 센서를 활용한 캘리브레이션 알고리즘을 구현하고자 한다. 또한, 추후 자동차 생산라인에서 센서 장착 검사 단계에 알고리즘 적용이 가능한 지 검토하고자 한다.

1.3 연구요약

본 연구의 알고리즘의 전체 흐름도와 알고리즘 흐름도는 다음과 같다.(그림1, 그림 2) 먼저, LiDAR 포인트 클라우드 상에서 clustering 기법을 이용하여 관심영역(ROI)의 3차원 좌표를 얻는다. 이때, RANSAC 알고리즘[8]을 이용하여 큐브의 세 평면을 검출한 후 outlier를 제거하고, inlier만을 남긴다. 얻은 inlier 정보에서 Plane Fitting을 통해 큐

브의 3평면을 얻고, 그에 따른 법선벡터를 추출한다. 각 평면의 각도가 90° 가 되도록, 비선형방정식을 풀어 법선 벡터의 각 parameter 값을 구한다. 큐브의 꼭짓점의 좌표는 세 평면의 교점과 법선 벡터, 큐브의 크기를 통해 구한다. Levenberg-Marquardt 방법[9]을 이용하여, LiDAR Ground Truth와 상대 위치 간의 변환 행렬을 얻을 수 있다.

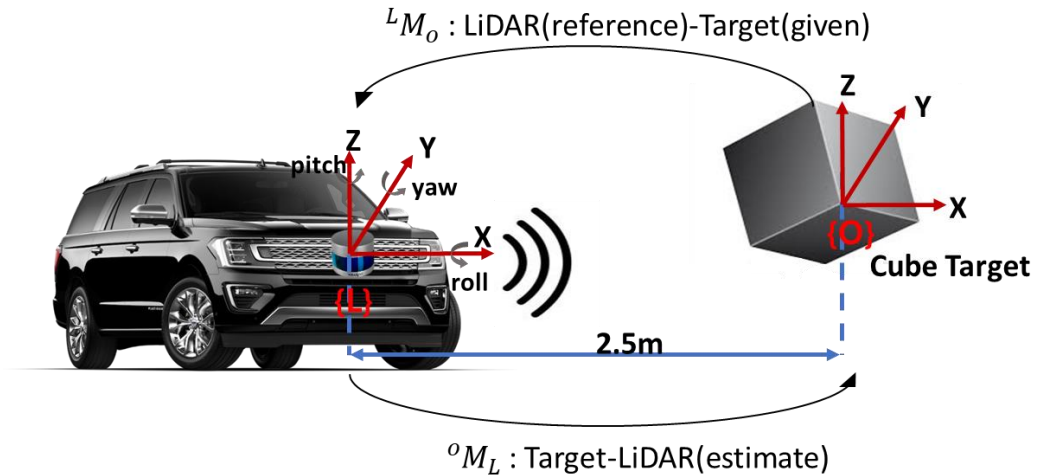


그림 1. 전체 흐름도

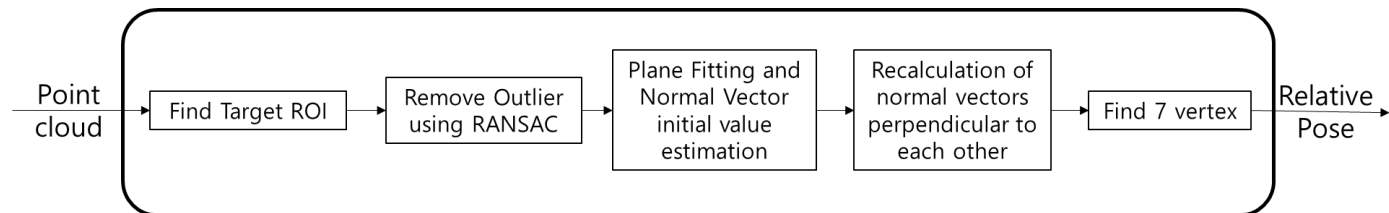


그림 1. 알고리즘 흐름도

2. 연구 방법

앞서 서론에서 제시한 것처럼, 평면 타겟의 경우 LiDAR의 FOV에 영향을 많이 받기에 특정한 조건에서는 평면 타겟 특징점의 추출이 불가능 하다. 하지만 3D 큐브 타겟을 사용하여 LiDAR와 타겟 간의 상대위치추정을 통해 평면 타겟에서 발생했던 LiDAR의 낮은 FOV에 대한 한계점을 극복 가능하다.

먼저, LiDAR를 통해 큐브 타겟에 대한 point cloud 데이터를 취득 후, 큐브의 특징점인 7개의 꼭짓점을 찾게 된다[3]. 3D 큐브 타겟의 특성을 이용하여, 각 면들의 평면의 방정식을 구하고 세면이 직교하는 지점을 특징점으로 추출할 수 있다. 이 특징점을 통하여 평면의 방정식과 큐브의 사이즈를 알고 있기에 나머지 6개의 특징점을 추출할 수 있다.

추출한 7개의 특징점을 LiDAR 좌표계를 기준으로 reference로 지정한 후, 실제 큐브 타겟의 특징점을 추출하여 reference 큐브와 실제 큐브 간의 상대위치추정을 통해 6-DOF에 대한 거리 및 각도의 오차측정이 가능하다.

앞선 내용을 바탕으로 구현한 알고리즘을 시뮬레이션과 실험에 대해 translation과 각도를 변화시키며 상대위치를 추정하여 알고리즘이 유효한지 검증할 것이다.

2.1 LiDAR point cloud 상의 큐브 타겟 특징점 추정

2.1.1 좌표계 정의 및 flow chart

본 연구는 LiDAR와 큐브 타겟 간의 상대위치를 추정함으로써 LiDAR장착 오류를 검사하게 된다. 그러기 위해서 LiDAR와 타겟 간의 상대위치 정보와 좌표계 정의가 이루어져야 한다.

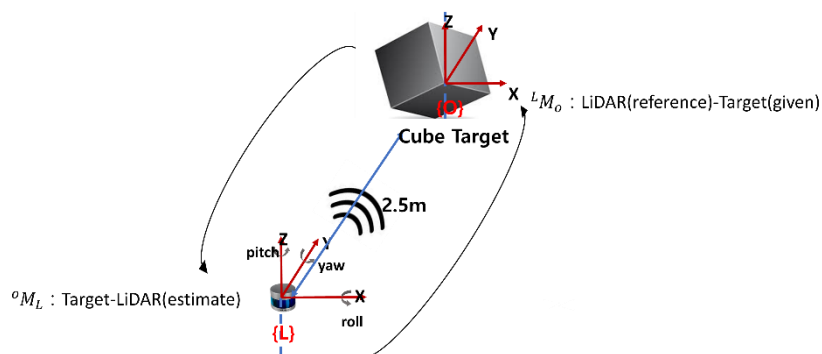


그림 3. LiDAR, 큐브 타겟의 좌표계 정의

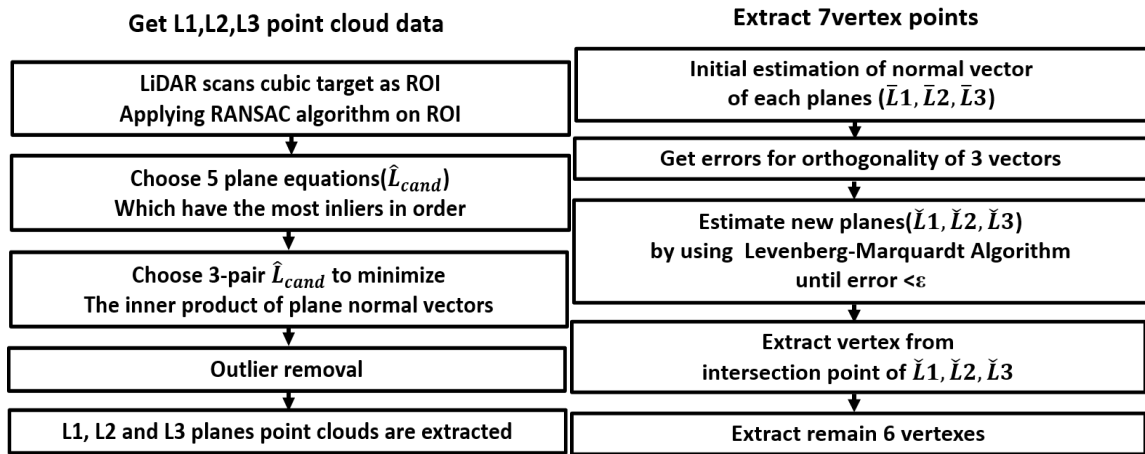


그림 4. 큐브 타겟의 특징점 추정 알고리즘

위의 그림4에서 제시하는 알고리즘은 포인트 클라우드 데이터를 특징점으로 사용하지 않고 꼭짓점 7개를 추정 가능하다는 장점이 있다. 이를 통해 LiDAR의 낮은 FOV에 관한 한계점을 극복 가능하다. 또한 상대위치추정을 통해 LiDAR 장착오차를 정확하게 검증하는데 있어 가장 불확실성 요소인 RANSAC의 랜덤성을 극복하기 위한 방법들이 위의 알고리즘에서 제시되어 있다.

2.1.2 ROI detecting 및 RANSAC을 통한 포인트 클라우드 클러스터링

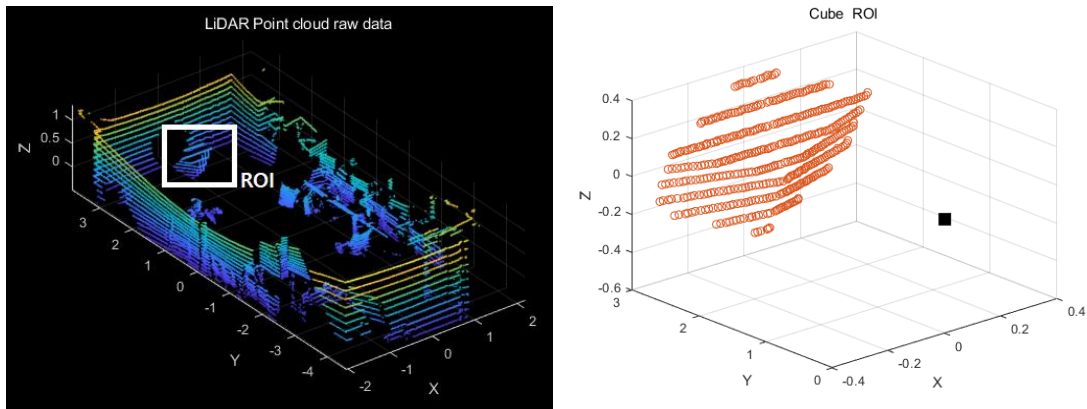


그림 5. LiDAR raw data 및 ROI

먼저 그림5에서 볼 수 있듯이 전체 LiDAR raw data에서 ROI(Region Of Interest)를 추출해야 한다. 클러스터링 기법을 통해 포인트 클라우드 데이터를 분할하고 큐브 타겟의 데이터 개수 특성과 y축 거리 정보를 이용하여 타겟 큐브를 ROI데이터로 추출하였다.

이후 추출된 ROI의 포인트 클라우드 데이터에 Sequential RANSAC 알고리즘을 적용하여 데이터 중 무작위로 선정된 세 점으로 평면 방정식을 구한다. 동일한 평면이 선정되는 것을

방지하기 위하여, 두 평면의 법선 벡터가 이루는 각이 $\pm 10^\circ$ 이내인 경우는 동일한 평면으로 간주하여 한 평면은 제외시키고, 설정한 iteration만큼의 과정을 반복 후 inlier가 가장 많은 상위 평면 5개를 선정한다. 선정 기준은 평면과 LiDAR포인트의 거리가 0.02[m] 이내인 점으로 설정한다.

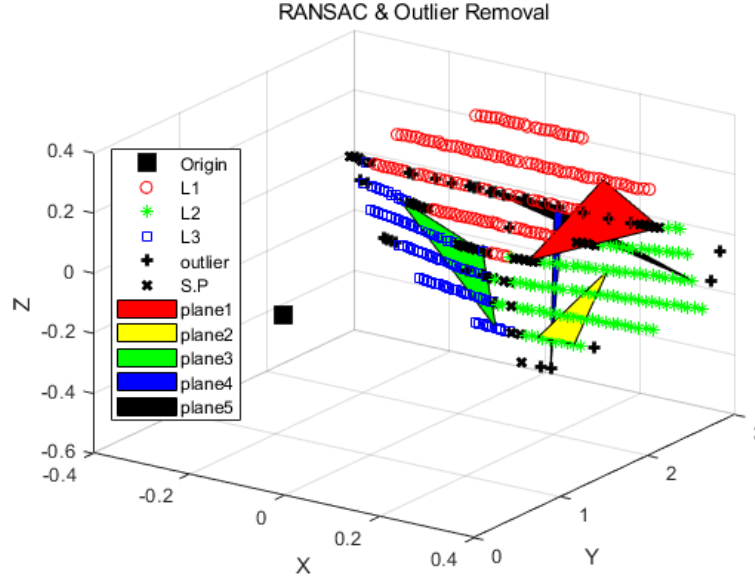


그림 6. LiDAR raw data 및 ROI

그림6에서 확인할 수 있듯, 선정된 5개의 평면 중, 수직한 세 평면쌍을 찾기 위해 각 평면의 법선 벡터 사이의 내적의 합 E를 최소화하는 평면쌍을 식(1)을 통해 찾는다.

$$E = |n_1^T n_2| + |n_2^T n_3| + |n_3^T n_1| \quad (n_i \text{ normal vector to } L_i) \quad (1)$$

이를 통해 추출된 세 평면쌍 $\hat{L}_{1,2,3}$ 에 대하여 Outlier를 제거하는 과정을 거치게 된다. 이후 Outlier를 제거한 포인트 클라우드를 다시 클러스터링 한다. 이 클러스터링 된 데이터를 통해 다시 plane fitting과정을 거쳐야 하므로, 수직에 가까운 평면을 추출하기 위해 각 평면쌍 $\hat{L}_{1,2,3}$ 이 공유하고 있는 포인트 클라우드 데이터도 이상치로 가정하여 제거한 후 최종적으로 $L_{1,2,3}$ 에 대한 포인트 클라우드를 클러스터링 한다.

2.1.3 Plane fitting 및 수직한 세 평면 방정식 추출

2.1.2에서 찾은 포인트 클라우드 데이터를 이용하여 각 $L_{1,2,3}$ 데이터로부터 $Residual^2$ 이 최소가 되는 평면의 방정식 세 쌍을 찾는다. 식(2)

$$\sum_{i=1}^n r_i^2 = \sum_{i=1}^n (a_k x_i + b_k y_i + c_k z_i)^2 \quad (2)$$

(where $k = 1,2,3$ which $L_{1,2,3}$ normal vector parameter,
 $n = \text{number of each } L_{1,2,3} \text{ point cloud data}$)

하지만 이렇게 얻은 세 평면 방정식은 각각이 완벽한 수직을 이루지 못하며, 이러한 점은 이후 상대위치추정과정에서 창작오차의 정밀도와 정확도를 떨어트리는 원인 중 하나이다. 그렇기에 완벽히 수직인 세 평면쌍을 찾기 위해 비선형 방정식 Solver인 Levenberg-Marquardt 알고리즘을 적용한다. 식(3)에서 볼 수 있듯이 한 평면은 고정시키고 나머지 평면을 두 평면의 방정식이 이루는 각이 0° 가 될 때까지 조정하여 수직인 평면의 방정식을 얻어낸다.

$$\begin{aligned} \text{Cost Function } F(\beta) &= \frac{1}{2} \sum \|a_f a_c + b_f b_c + c_f c_c\|, \beta = (a_c, b_c, c_c) \\ (\text{Fixed plane} &= a_f x + b_f y + c_f z, \quad \text{compare plane} = a_c x + b_c y + c_c z) \\ \beta' &= \text{argmin} F(\beta) = \beta - \eta (J^T J + \lambda \text{diag}(J))^{-1} J^T F(\beta) \\ \beta' &= (a'_c, b'_c, c'_c) \end{aligned} \quad (3)$$

이를 통해 얻어진 β' 는 기존 비교 평면인 $a_c x + b_c y + c_c z = 0$ 를 고정된 평면 $a_f x + b_f y + c_f z = 0$ 에 대하여 최적으로 수직인 평면 방정식이다. 이후 L_1 을 고정시키고 L_2 을 비교하여 L'_2 을 얻는다. L'_2 을 고정시킨 후 L_3 를 비교한 후 얻어진 L'_3 를 다시 L_1 과 비교함으로써 L''_3 을 얻어 최종적으로 서로 완벽히 수직인 세 평면의 방정식 L_1, L'_2, L''_3 을 얻게 된다.

$$\begin{aligned} &\text{Fixed plane} \quad \text{Compare plane1} \quad \text{Compare plane 2} \quad \text{Orthogonal Plane} \\ &\quad (L_1 \ L_1 \ L_2 \ L_2 \ L_3 \ L_3) \quad (L_2 \ L_3 \ L_1 \ L_3 \ L_1 \ L_2) \rightarrow \\ &\quad (L'_2 \ L'_3 \ L'_1 \ L'_3 \ L'_1 \ L'_2) \quad (L_3 \ L_2 \ L_3 \ L_1 \ L_2 \ L_1) \rightarrow \\ &\quad (L''_3 \ L''_2 \ L''_3 \ L''_1 \ L''_2 \ L''_1) \quad (P_1 = [L_1; L'_2; L''_3] \ P_2 = [L_1; L''_2; L'_3] \ P_3 = \\ &\quad [L'_1; L_2; L''_3] \ P_4 = [L''_1; L_2; L'_3] \ P_5 = [L'_1; L''_2; L_3] \ P_6 = [L''_1; L'_2; L_3]) \\ &\quad \text{Final Plane} \\ &\quad (\hat{L}_1 \ \hat{L}_2 \ \hat{L}_3) \rightarrow \frac{P_1 + P_2 \dots + P_6}{6} \end{aligned} \quad (4)$$

이러한 과정을 식(4)에서 보는 것처럼 각 $L_{1,2,3}$ 의 모든 경우에 적용하여 구한 후 평균한 값으로 최종 $\hat{L}_{1,2,3}$ 의 평면 방정식을 구할 수 있다.

2.1.4 큐브 타겟 특징점 추출

2.1.3에서 구한 서로 수직인 평면 방정식 $\hat{L}_{1,2,3}$ 을 알고 있으므로, 각 평면이 교차하는 곳에서 특징점을 찾을 수 있다. 교점의 특징점을 구하기 위해, \hat{L}_1 의 평면의 임의의 점을 \hat{L}_2 로 사영시키고, 다시 \hat{L}_3 으로 사영시킨 후 이 과정을 \hat{L}_2, \hat{L}_3 에 대해서도 시행하여 교점의 후보를 구한 후, 평균한 값으로 교점에서의 특징점을 추출할 수 있다. 이후 기존에 알고 있던 법선 벡터와 큐브 타겟의 스펙을 활용하여 식(5)를 통해 나머지 특징점에 대해서도 추정할 수 있다.

$$\begin{aligned} \text{Vertex}_{2,4,6} &= \text{Vertex7} + \text{cube scale} \times L_{3,1,2} \\ \text{Vertex}_{3,5,1} &= \text{Vertex2,6,6} + \text{cube scale} \times L_{1,1,3} \end{aligned} \quad (5)$$

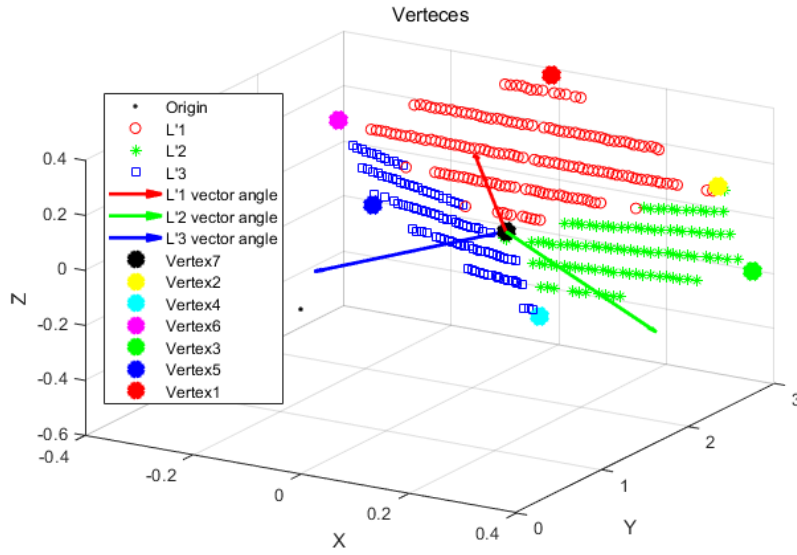


그림 7. 큐브 타겟 특징점 추출

2.2 Reference와 타겟간의 상대위치 추정

2.2.1 Reference 정의

상대위치추정을 통한 장착오차 검증을 위해선, 앞서 2.1과정을 통해 얻은 특징점들을 (0,0,0)에 큐브 센터를 두고 LiDAR 좌표축에 알고리즘 반복 횟수 n 만큼 각 특징점을 평균 낸 값으로 특징점을 투영시켜 가상의 reference 큐브를 만든다. 식(6)

$$OVertexK = \frac{\sum_{n=1}^{iter} Vertex_n K}{n} \text{ (which } K = 1, 2, \dots, 7) \quad (6)$$

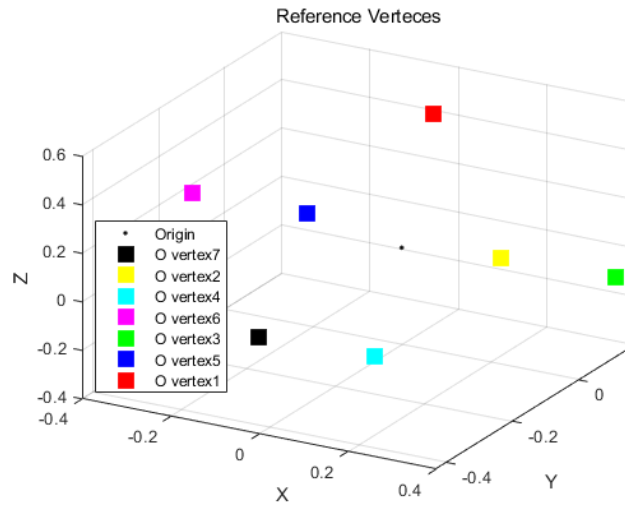


그림 8. Reference 큐브 특징점

2.2.2 Reference – Target 간의 상대위치추정

2.2.1에서 구한 reference와 타겟 큐브 간의 상대위치추정은 비선형 방정식 Solver 인 Levenberg-Marquardt 알고리즘을 통해 얻어진다. 6-DOF에 대하여 상대위치추정을 위한 Transformation matrix M은 Rotation인 Roll, Pitch, Yaw와 Translation인 $\Delta X, \Delta Y, \Delta Z$ 에 대해 식 (7)과 같이 정의된다.

$$\begin{aligned} LOR &= R_z(\phi)R_y(\theta)R_x(\psi) = \\ & \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & \sin(\phi) & \cos(\phi) & 0 & 0 & 1 \\ \sin(\phi) & \cos(\phi) & 0 & \cos(\phi) & -\sin(\phi) & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \cos(\psi) & -\sin(\psi) \\ 0 & 0 & 0 & 1 & 0 & 0 & \sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 & 1 & 0 \\ 0 & \cos(\theta) & 0 & \sin(\theta) & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \\ LOT &= [\Delta X, \Delta Y, \Delta Z]^T \\ LOM &= [LOR|LOT] \end{aligned} \quad (7)$$

이러한 변환행렬 LOM는 Reference의 특징점인 OP에 대하여 LiDAR를 통해 감지되고 있는 타겟 큐브의 특징점인 LP의 상대위치를 구할 수 있게 된다. 식(8)

$$OP = LOM LP = [LOR|LOT] LP \quad (8)$$

특징점의 개수는 7개 이므로, 각 OP_i, LP_i (which $i = 1, 2, \dots, 7$) 에 대해서 Levenberg-Marquardt 알고리즘을 적용하여 상대위치추정을 하게 된다. 비선형 방정식을 풀기위한 Cost Function $F(\beta)$ 는 식(9) 와 같이 정의된다.

$$\begin{aligned} F(\beta) &= \frac{1}{2} \sum \| OP - [LOR|LOT] LP \|, \quad \beta = (\phi, \theta, \psi, \Delta x, \Delta y, \Delta z) \\ LP &= [LVertex1^T, LVertex2^T, \dots, LVertex7^T], \\ OP &= [OVertex1^T, OVertex2^T, \dots, OVertex7^T] \end{aligned} \quad (9)$$

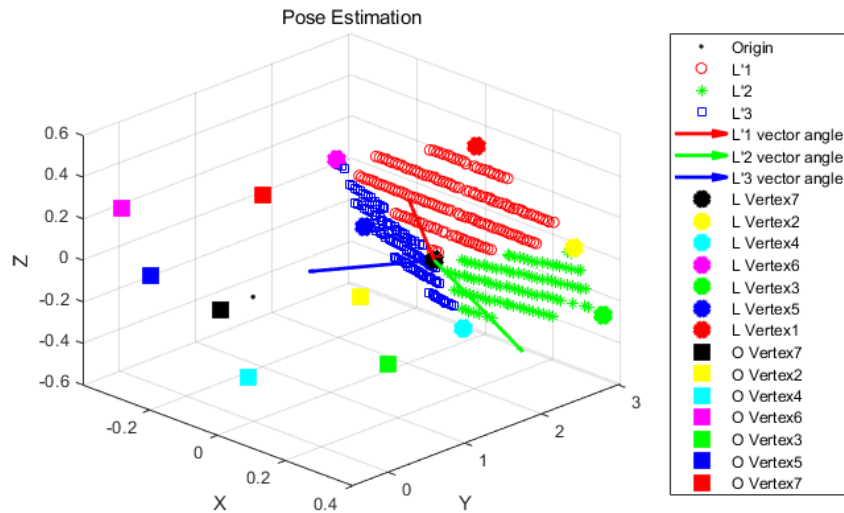


그림 9. 큐브 상대위치추정

Levenberg-Marquardt 알고리즘에 Cost Function을 대입하여 식(10)을 통해 얻어진 β^* 는 OP

에 대한 LP 의 상대위치 에러가 ε 미만이 되는 최적의 해이다. 이를 통해 LP 의 상대위치를 추정 및 LiDAR장착오류를 검증할 수 있다.

$$\beta^* \equiv \operatorname{argmin} F(\beta)$$

$$\beta^* = \beta - \eta (J^T J + \lambda \operatorname{diag}(J))^{-1} J^T F(\beta)$$

(where J is the Jacobian matrix for cost function F , $\eta=0.02$, $\lambda=0.3$) (10)

3. 연구 결과

3.1 시뮬레이션 결과

3.1.1 시뮬레이션 환경

LiDAR와 카메라의 가상 데이터를 수집하기 위해 “Blender” 프로그램을 사용하여 그림 10와 같이 시뮬레이션 환경을 구상하였다. 각 센서와 타겟은 ADAS 탑재 차량의 경우를 가정하여 배치했다. Ground Truth는 표 1와 같이 설정했으며 타겟으로부터 각 센서 간의 거리는 2.5[m]이다. 각 센서의 규격은 표 1과 같다.

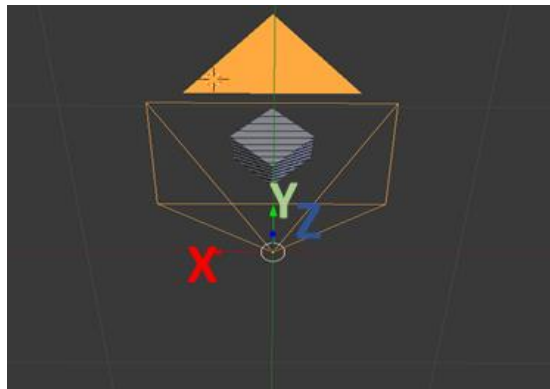



그림 20. Blender 시뮬레이션 구상

표 1. 시뮬레이션 환경 센서 규격

Sensor Specification				Virtual Sensor
LiDAR (Velodyne HDL 32E)	Scan resolution	Noise mean (Depth)	Noise σ (Depth)	
	0.17 [deg]	0.00 [m]	0.02 [m]	

3.1.2 시뮬레이션 결과

상기의 시뮬레이션 환경에서 얻게 된 가상의 Velodyne HDL 32E 모델의 포인트 클라우드 데이터를 통해 본 연구의 알고리즘을 적용하여 표와 그림과 같이 상대위치 추정 결과를 도출하였다. 테스트는 크게 2가지 경우에 대해서 진행되었다. X축 이동과 Z축 회전에 대한 상대위치추정 오프셋 및 오차에 대해 구하여 이상적인 상황과 비교하였다. X축 이동의 경우 $-30\text{mm} \sim 30\text{mm}$ 까지 5mm 의 step으로 움직이며 위치를 추정하였고, Z축 회전의 경우는 $-3^\circ \sim 3^\circ$ 까지 0.5° 의 step으로 움직이며 위치를 추정하였다.

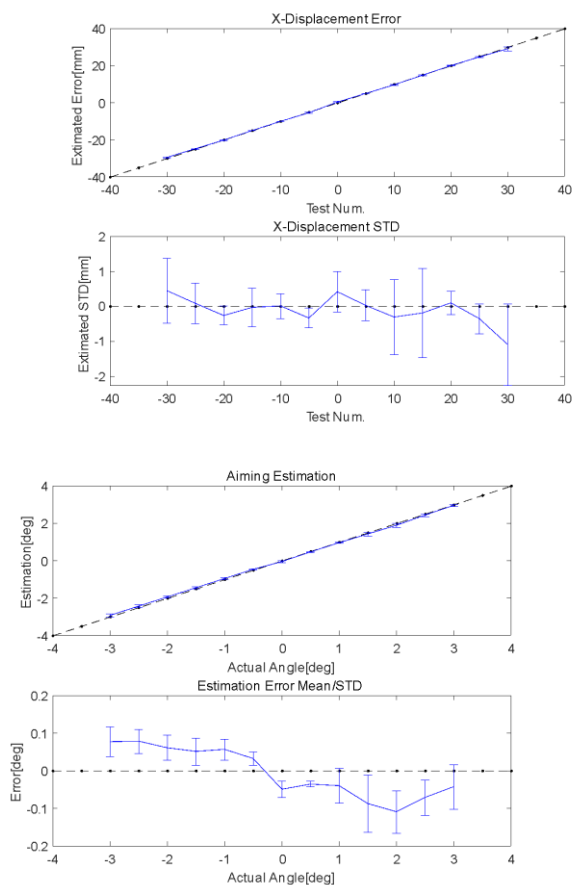


그림 11. 시뮬레이션 상대위치 추정 결과

표 2. 시뮬레이션 상대위치 추정 결과

Error	Translation [mm]		Rotation [deg]	
	mean	max	mean	max
Mean	-0.1142mm	0.4453mm	-0.0059°	0.0780°

STD	0.6350mm	1.2783mm	0.0384°	0.0757°
-----	----------	----------	---------	---------

3.2 실험 결과

3.2.1 실험 환경 구성

그림 15와 같이 실험 환경을 구성하였다. 타겟으로부터 각 센서 간의 거리는 2.5[m]이다. 알고리즘 검증에 위해 각 센서의 절대위치 간의 차이인 Ground Truth와 알고리즘을 통해 추정된 상대위치를 비교하였다. LiDAR를 x축으로 $\pm 30[\text{cm}]$ 을 $\pm 5[\text{cm}]$ 간격으로 평행 이동한다. y축으로 $\pm 3^\circ$ 를 $\pm 0.5^\circ$ 간격으로 회전 이동한다. 각 x축 y축으로 이동하며 동일한 큐브 타겟물에 대한 캘리브레이션 연구를 진행하였다. 실험을 위해 사용한 LiDAR 규격은 표3와 같다.

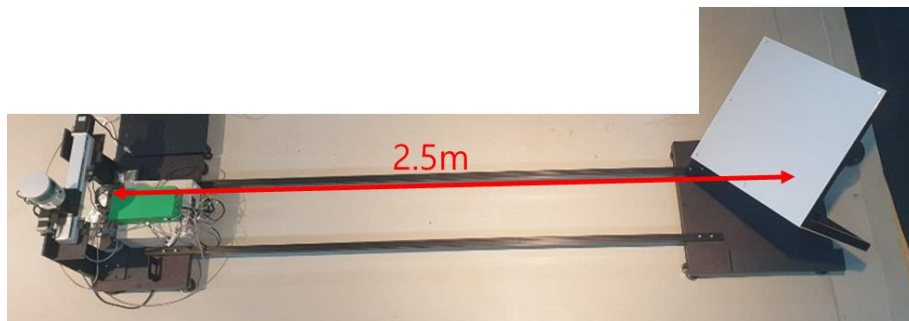



그림 12. 실험 환경 구성

표 2. 실험 환경 센서 규격

Sensor Specification				Sensor Image
LiDAR (Velodyne VLP-16)	VER./HORIZ FOV	Range Accuracy	VER./HORIZ Angular Resolution	
	$\pm 15^\circ/360^\circ$	$\pm 3 [\text{cm}]$	0.02 [m]	

3.2.2 실험 결과

앞선 시뮬레이션 테스트 조건과 동일한 조건으로 실험을 진행하였다. 실제 실험환경

의 노이즈를 고려하여 50개의 데이터 셋에 대한 실제 LiDAR위치와 상대위치추정 결과의 오차 평균치 및 분산을 표2에 나타냈고, 그림 을 통해 경향성을 확인할 수 있다. 표3을 통하여서는 시뮬레이션 대비 실험결과와 평면 타겟을 기반으로 한 상대위치추정 결과와 비교하였다.

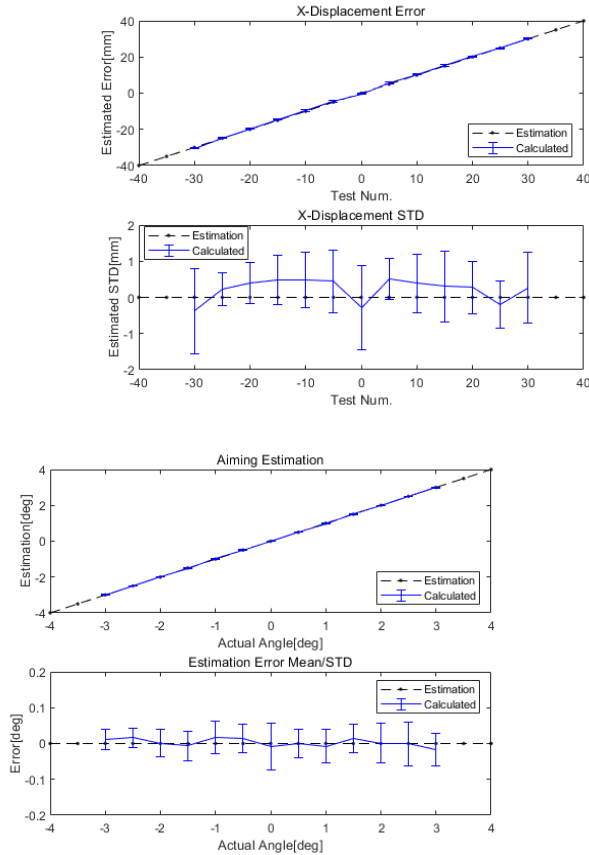


그림 13. 실험 결과

표 4. 실험 결과

Error	Translation [mm]		Rotation [deg]	
	mean	max	mean	max
Mean	-0.2248mm	0.5224mm	-0.0025°	0.0168°
STD	0.8021mm	1.1848mm	0.0441°	0.0644°

표 5. 실험 결과

Error	Translation [mm]		Rotation [deg]	
	mean	EXP mean	mean	EXP mean
Simulation mean	-0.1142mm	-0.2248mm	-0.0059°	-0.0025°
Plane mean	2.6000mm		0.0097	
Simulation STD	0.6350mm	0.8021mm	0.0384°	0.0441°

Plane STD	5.1400mm		0.0961	
-----------	----------	--	--------	--

4. 토 의

4.1 시뮬레이션 결과 분석

시뮬레이션 프로그램인 Blensor 프로그램에서는 실제 실험에서 사용하는 LiDAR인 Velodyne VLP-16 모델보다 Vertical로 FOV가 2배 높은 Velodyne HDL 32E 모델을 사용하여 시뮬레이션을 진행해야 했다는 한계점이 있다. 실제 실험 환경과 비슷한 환경조성을 위해 취득한 포인트 클라우드 데이터에 Gaussian noise($\sigma=0.02[m]$)을 인가하여 알고리즘에 대입하였다.

시뮬레이션 결과를 그림, 표를 통해 확인하였을 경우, LiDAR의 상대위치에 따라 X축 이동과 Z축 회전에 대해 경향성을 잘 추종하는 것을 확인할 수 있었다. 표를 통해서 X축 이동에 대해선 평균 $-0.1142mm$ 의 정확도를 보여주며 $0.6350mm$ 의 정밀도를 보여주었다. Z축 회전에 대해서는 평균 -0.0059° 의 정확도를 보여주며 0.0384° 의 정밀도를 보여주었다. 알고리즘을 통해 구한 오차 값들이 충분히 허용할 만한 오차범위 이내이며, 이를 통해 알고리즘이 LiDAR 포인트 클라우드 데이터에 대해 작동하는지에 대해 유효성을 검증할 수 있었다.

4.2 실험 결과 분석

그림의 실험 환경에서 Motion stage를 통해 각 테스트의 pose별로 50개의 dataset에 대해 알고리즘을 통하여 상대위치를 추정해 실제 위치에 대한 오차의 평균치를 얻었다.

그림을 통해 실험에서 LiDAR의 상대위치에 따라 X축 이동과 Z축 회전에 대해 경향성을 잘 추종하는 것을 확인할 수 있었다. 표를 통해서 X축 이동에 대해선 평균 $-0.2248mm$ 의 정확도를 보여주며 $0.8021mm$ 의 정밀도를 보여주었다. Z축 회전에 대해서는 평균 -0.0025° 의 정확도를 보여주며 0.0441° 의 정밀도를 보여주었다. 또한 표를 통해 시뮬레이션과 평면 타겟을 기반으로 한 상대위치추정 연구 결과값과의 차이에 대해서도 비교할 수 있다. X축 이동과 Z축 회전에 대한 결과에선 실제 실험환경임에도 불구하고 시뮬레이션과 실험의 결과가 큰 차이를 보이지 않았다. 하지만 평면 타겟과의 실험 결과를

비교하였을 때, Z축 회전은 물론, 특히 X축 이동에 대해서 정확성과 정밀성에 있어 우수한 성능을 보여주었다.

초기 이 연구의 특징에서 타겟의 특징점에 라이다 포인트를 사용하지 않음으로써 저사양 LiDAR모델의 낮은 FOV를 극복할 수 있는 알고리즘을 제시하였다. 라이다 포인트를 특징점으로 사용하는 평면 타겟과의 상대위치추정 오차에 있어 본 연구가 제시하는 큐브 타겟에 대한 알고리즘이 훨씬 뛰어난 성능을 보여주었고, 시뮬레이션 상에서 실험 환경에 사용하는 LiDAR보다 높은 FOV의 LiDAR를 사용하여 상대위치추정 오차를 구했음에도, 실제 실험 결과와 큰 차이가 없다는 점에서도 연구 목적에 부합하는 알고리즘의 성능을 확인할 수 있었다.

하지만, Motion stage에서 하드웨어적으로 생기는 축 회전에 대한 커플링 문제에 의해 X축 회전에 대해서는 잘 추종하지 않는 한계점을 보여주었다.

5. 결 론

본 연구에서는 저해상도 라이다와 RGB 카메라를 이용하여 캘리브레이션을 수행하는 알고리즘을 구현하였다. 일반적인 캘리브레이션 방식인 평면 보드를 타겟으로 사용하지 않고 3차원 큐브를 타겟으로 사용하여 캘리브레이션을 수행하였다.

각 센서의 데이터로부터 큐브 타겟의 특징점인 7개의 꼭짓점을 추출하였고, PnP의 솔루션으로는 각각의 좌표를 매칭하여 상대위치와 변환행렬을 얻는 EPnP 알고리즘을 사용하였다. 특히 LiDAR 포인트의 특징점 추출 시, 두 번의 RANSAC 알고리즘을 수행하며, 이상치를 제거하는 과정을 반복하였다. 이를 통해 특징점의 좌표를 더 정확하게 추정할 수 있었다. 알고리즘을 수행하여 얻은 결과는 센서 간 상대위치와 각 센서 데이터의 좌표계를 동기화하기 위한 변환행렬이며, 센서융합 알고리즘 구현 시 필수적인 정보이다.

본 연구의 알고리즘을 통해 추정한 변환행렬은 LiDAR 3D 데이터를 카메라 2D 이미지에 매핑하여 사용될 수 있다. 이를 활용하면 센서 부착 차량의 이동 환경에서, 변환행렬과의 연산을 수행하여 실시간으로 주행 환경을 인지하고 주변 환경을 파악할 수 있을 것이다. 또한, 생산라인과 같은 정지된 환경에서 알고리즘을 통해 추정한 상대위치를 사용하면, 센서의 Ground Truth 값과 실제 부착량 사이의 오차를 파악할 수 있을 것이다.

6. References

- [1] Atanacio-Jiménez, G., González-Barbosa, J. J., Hurtado-Ramos, J. B., Ornelas-Rodríguez, F. J., Jiménez-Hernández, H., García-Ramírez, T., & González-Barbosa, R. (2011). Lidar velodyne hdl-64e calibration using pattern planes. *International Journal of Advanced Robotic Systems*, 8(5), 59.
- [2] Y. Park, S. Yun, C. S. Won, K. Cho, K. Um, and S. Sim. “Calibration between color camera and 3d lidar instruments with a polygonal planar board”. *Sensors*, 14(3):5333–5353, 2014. 1, 2, 3, 5, 6
- [3] M.Velas, M.Spanel, Z.Materna, and A.Herout. Calibration of rgb camera with velodyne lidar. In *WSCG 2014 Communication Papers Proceedings*, volume 2014, pages 135–144. Union Agency, 2014. 1, 2
- [4] Geiger, A., Moosmann, F., Car, Ö., & Schuster, B. (2012, May). Automatic camera and range sensor calibration using a single shot. In *2012 IEEE International Conference on Robotics and Automation* (pp. 3936-3943). IEEE.
- [5] Kim, E. S., & Park, S. Y. (2019, July). Extrinsic calibration of a camera-LIDAR multi sensor system using a planar chessboard. In *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)* (pp. 89-91). IEEE.
- [6] Zoltan Pusztai and Levente Hajder. “Accurate Calibration of LiDAR-Camera Systems using Ordinary Boxes”, *IEEE International Conference on Computer Vision Workshops*, 2017
- [7] You, J. H., Oh, S. T., Park, J. E., Eskandarian, A., & Kim, Y. K. (2020). Automatic LiDAR Extrinsic Calibration System using Photodetector and Planar Board for Large-scale Applications. *arXiv preprint arXiv:2008.10542*.
- [8] Derpanis, K. G. (2010). Overview of the RANSAC Algorithm. *Image Rochester NY*, 4(1), 2-3.
- [9] Moré, J. J. (1978). The Levenberg-Marquardt algorithm: implementation and theory. In *Numerical analysis* (pp. 105-116). Springer, Berlin, Heidelberg.