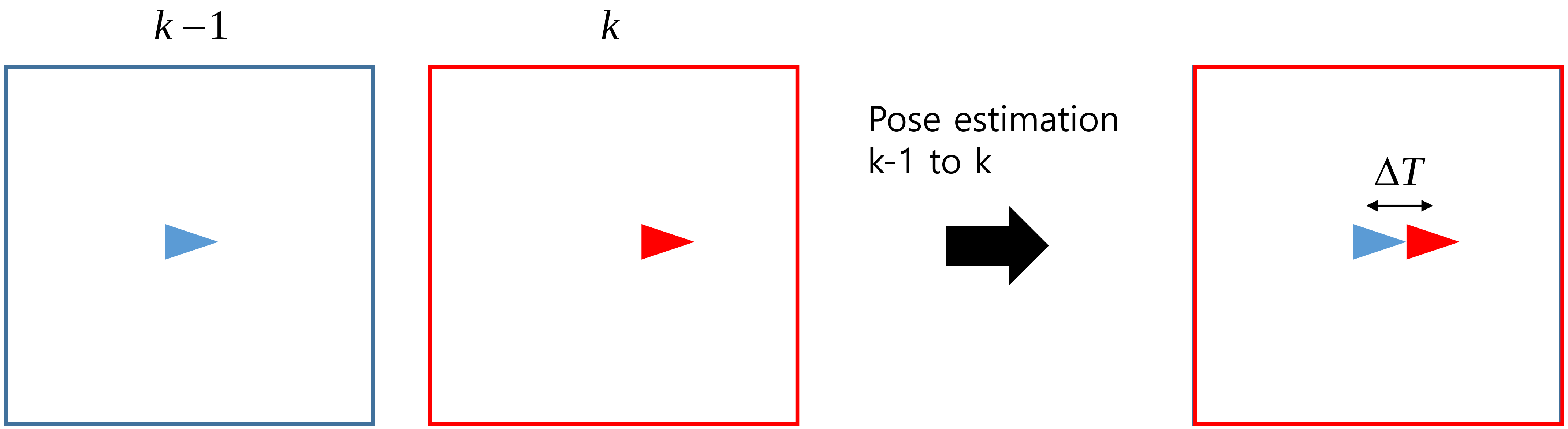
Velodyne SLAM

2017.2.13

김태원

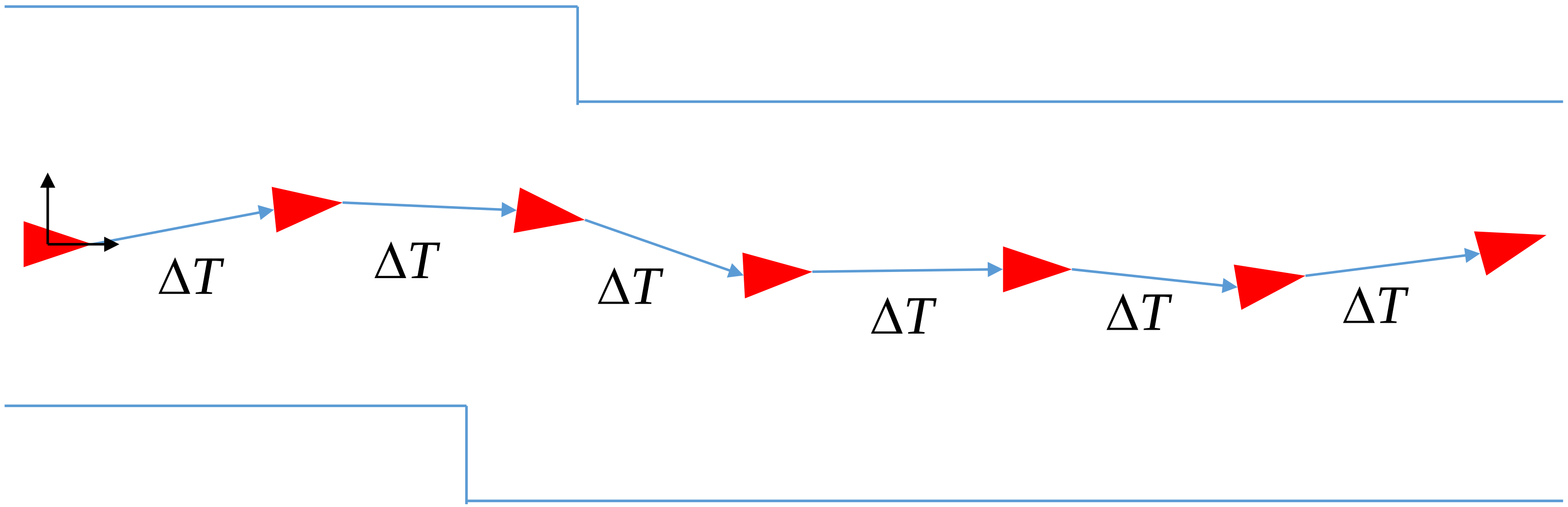
Overview

- What is SLAM(Simultaneous Localization and Mapping)



Overview

- What is SLAM(Simultaneous Localization and Mapping)



The KITTI Vision Benchmark Suite

A project of Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago

Karlsruhe Institute of Technology

home setup stereo flow scene flow odometry object tracking road semantics raw data submit results jobs

Andreas Geiger (MPI Tübingen) | Philip Lenz (KIT) | Christoph Stiller (KIT) | Raquel Urtasun (University of Toronto)

Welcome to the KITTI Vision Benchmark Suite!

We take advantage of our autonomous driving platform Annieway to develop novel challenging real-world computer vision benchmarks. Our tasks of interest are: stereo, optical flow, visual odometry, 3D object detection and 3D tracking. For this purpose, we equipped a standard station wagon with two high-resolution color and grayscale video cameras. Accurate ground truth is provided by a Velodyne laser scanner and a GPS localization system. Our datsets are captured by driving around the mid-size city of Karlsruhe, in rural areas and on highways. Up to 15 cars and 30 pedestrians are visible per image. Besides providing all data in raw format, we extract benchmarks for each task. For each of our benchmarks, we also provide an evaluation metric and this evaluation website. Preliminary experiments show that methods ranking high on established benchmarks such as Middlebury perform below average when being moved outside the laboratory to the real world. Our goal is to reduce this bias and complement existing benchmarks by providing real-world benchmarks with novel difficulties to the community.

Share

Select category: City | Residential | Road | Campus | Person | Calibration

Data Category: City

Before browsing, please wait some moments until this page is fully loaded.

2011_09_26_drive_0001 (0.4 GB)

Length: 114 frames (00:11 minutes)
Image resolution: 1392 x 512 pixels
Labels: 12 Cars, 0 Vans, 0 Trucks, 0 Pedestrians, 0 Sitters, 2 Cyclists, 1 Trams, 0 Misc
Downloads: [unsynced+unrectified data] [synced+rectified data] [calibration] [tracklets]

2011_09_26_drive_0002 (0.3 GB)

Length: 83 frames (00:08 minutes)
Image resolution: 1392 x 512 pixels
Labels: 1 Cars, 0 Vans, 0 Trucks, 0 Pedestrians, 0 Sitters, 2 Cyclists, 0 Trams, 0 Misc
Downloads: [unsynced+unrectified data] [synced+rectified data] [calibration] [tracklets]

2011_09_26_drive_0005 (0.6 GB)

Length: 160 frames (00:16 minutes)
Image resolution: 1392 x 512 pixels
Labels: 9 Cars, 3 Vans, 0 Trucks, 2 Pedestrians, 0 Sitters, 1 Cyclists, 0 Trams, 0 Misc
Downloads: [unsynced+unrectified data] [synced+rectified data] [calibration] [tracklets]

2011_09_26_drive_0009 (1.8 GB)

Length: 453 frames (00:45 minutes)
Image resolution: 1392 x 512 pixels
Labels: 89 Cars, 3 Vans, 2 Trucks, 3 Pedestrians, 0 Sitters, 0 Cyclists, 0 Trams, 1 Misc
Downloads: [unsynced+unrectified data] [synced+rectified data] [calibration] [tracklets]

Additional information used by the methods

- Stereo: Method uses left and right (stereo) images
- Laser Points: Method uses point clouds from Velodyne laser scanner
- Loop Closure Detection: This method is a SLAM method that detects loop closures
- Additional training data: Use of additional data sources for training (see details)

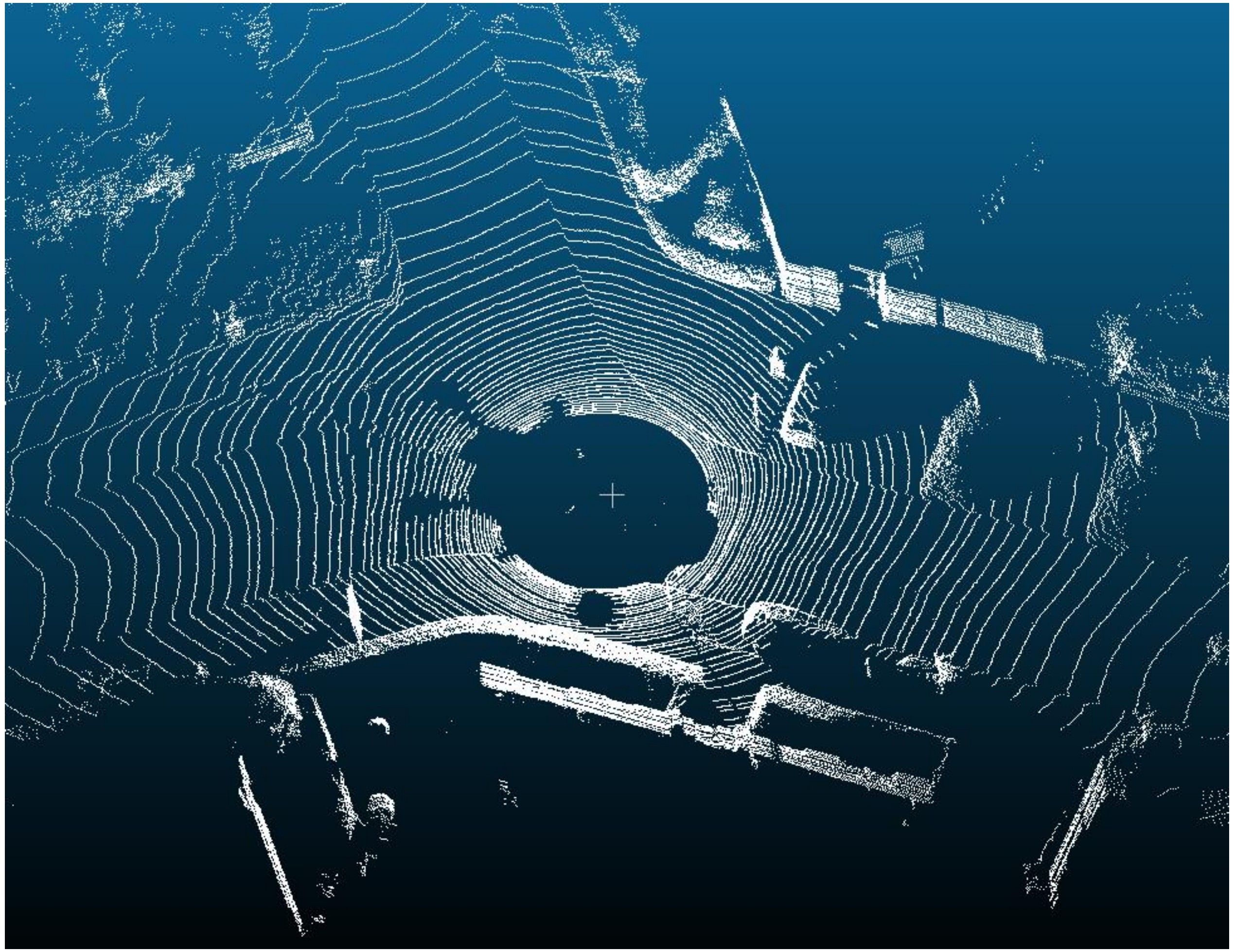
	Method	Setting	Code	Translation	Rotation	Runtime	Environment	Compare
1	V-LOAM	Laser Points		0.68 %	0.0016 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
J. Zhang and S. Singh: Visual-lidar Odometry and Mapping: Low drift, Robust, and Fast. IEEE International Conference on Robotics and Automation(ICRA) 2015.								
2	LOAM	Laser Points		0.70 %	0.0017 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
J. Zhang and S. Singh: LOAM: Lidar Odometry and Mapping in Real-time. Robotics: Science and Systems Conference (RSS) 2014.								
3	SOFT2	Stereo		0.81 %	0.0022 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
4	GDVO	Stereo		0.86 %	0.0031 [deg/m]	0.09 s	1 core @ >3.5 Ghz (C/C++)	
5	HypERROCC	Stereo		0.88 %	0.0027 [deg/m]	0.25 s	2 cores @ 2.0 Ghz (C/C++)	
6	SOFT	Stereo		0.88 %	0.0022 [deg/m]	0.1 s	2 cores @ 2.5 Ghz (C/C++)	
I. Cvišić and I. Petrović: Stereo odometry based on careful feature selection and tracking. European Conference on Mobile Robots (ECMR) 2015.								
7	RotRocc	Stereo		0.88 %	0.0025 [deg/m]	0.3 s	2 cores @ 2.0 Ghz (C/C++)	
M. Buczko and V. Willert: Flow-Decoupled Normalized Reprojection Error for Visual Odometry. 19th IEEE Intelligent Transportation Systems Conference (ITSC) 2016.								
8	EDVO	Stereo		0.89 %	0.0030 [deg/m]	0.1 s	1 core @ 2.5 Ghz (C/C++)	
9	svo2	Stereo		0.94 %	0.0021 [deg/m]	0.2 s	1 core @ 2.5 Ghz (C/C++)	

Overview

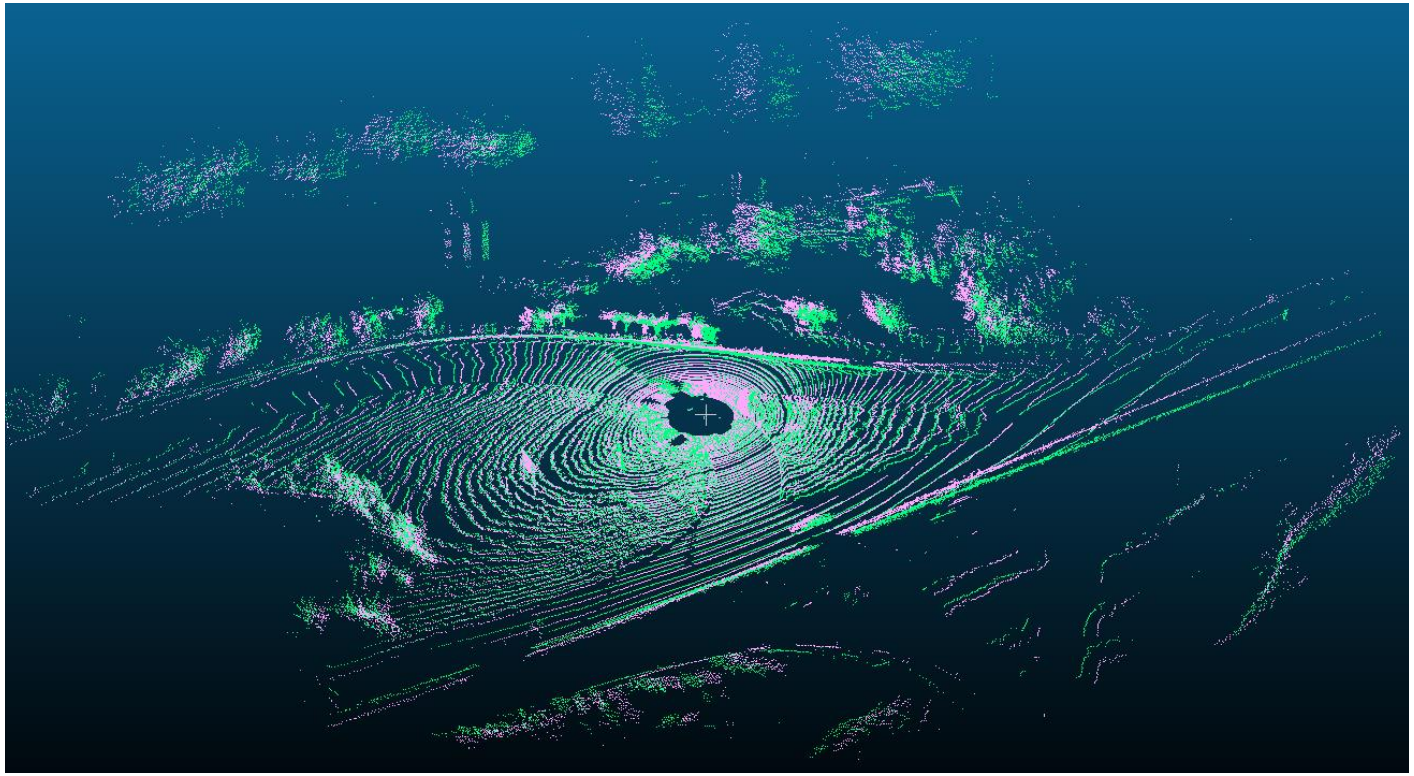
HDL-64E

KEY FEATURES

- 64 Channels
- 120m range
- 2.2 Million Points per Second
- 360° Horizontal FOV
- 26.9° Vertical FOV
- 0.08° angular resolution (azimuth)
- <2cm accuracy
- ~0.4° Vertical Resolution
- User selectable frame rate
- Rugged Design



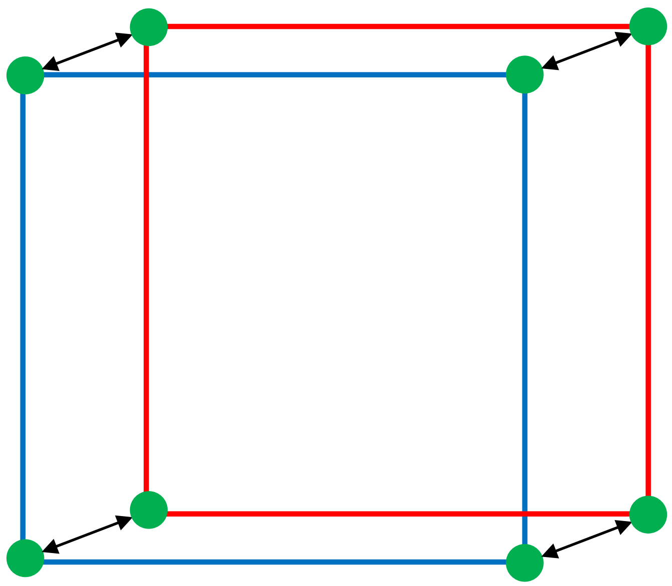
Overview



- Point cloud data의 특징
  - Set of local 3d points
  - Unspecified data
  - Non sorted data

[0]	{x=52.3221397 y=5.75702906 z=1.98781168 }
[1]	{x=78.8461075 y=12.7539253 z=2.90621853 }
[2]	{x=78.7890854 y=13.0015259 z=2.90511870 }
[3]	{x=58.2385941 y=10.2684298 z=2.20603037 }
[4]	{x=78.3154831 y=15.2258177 z=2.90221357 }
[5]	{x=77.3685989 y=15.2952299 z=2.87117505 }
[6]	{x=76.5581055 y=15.5135946 z=2.84607601 }
[7]	{x=77.6023254 y=16.4938755 z=2.88668418 }
[8]	{x=76.5940552 y=17.0413017 z=2.85744500 }
[9]	{x=76.6254425 y=17.1751499 z=2.85938954 }
[10]	{x=77.1300583 y=17.5455151 z=2.87824202 }

Overview

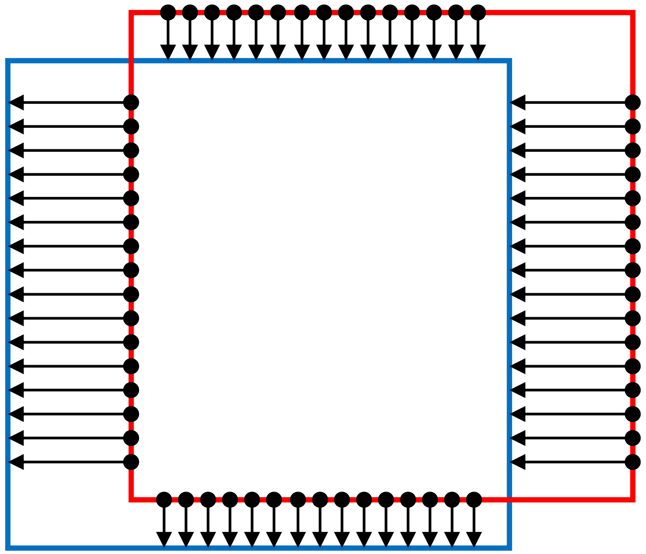


- Feature based method
  - Find matching pair
  - Minimize cost

$$\underset{\mathbf{T}}{\operatorname{argmin}}\left(\sum_{i}^{N}\left\|\mathbf{T}\mathbf{P}_{\mathbf{i}}^{\mathbf{k}}-\mathbf{P}_{\mathbf{i}}^{\mathbf{k-1}}\right\|^{2}\right)$$

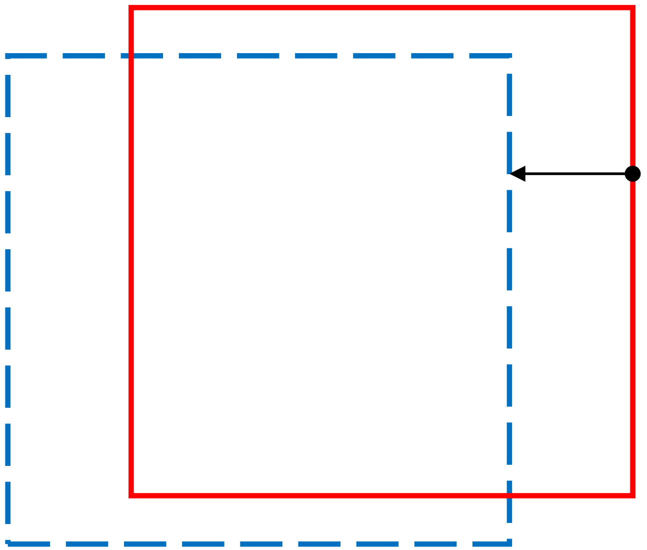
N: num of matching pair

Proposed method



- Minimize point to plane distance
  - No need matching pair

Proposed method

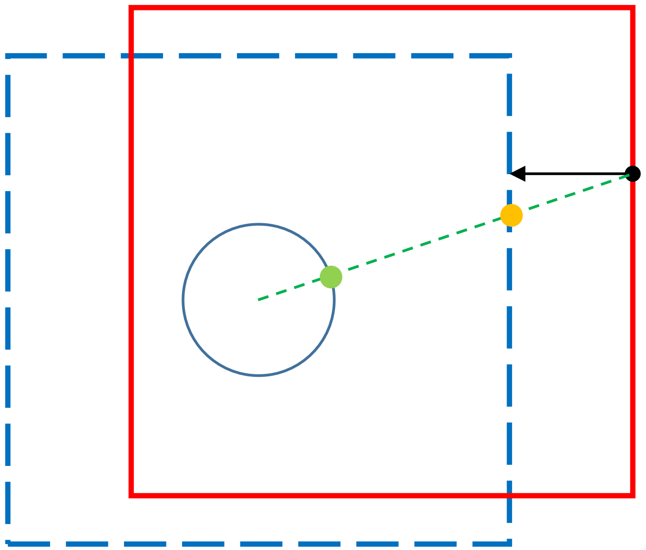


- Iterative closest point(ICP)

  1. Find nearest plane
  2. Calculate point to plane distance

  Search problem!! -> too slow
  Non-differentiable!! -> Heuristic method

Proposed method

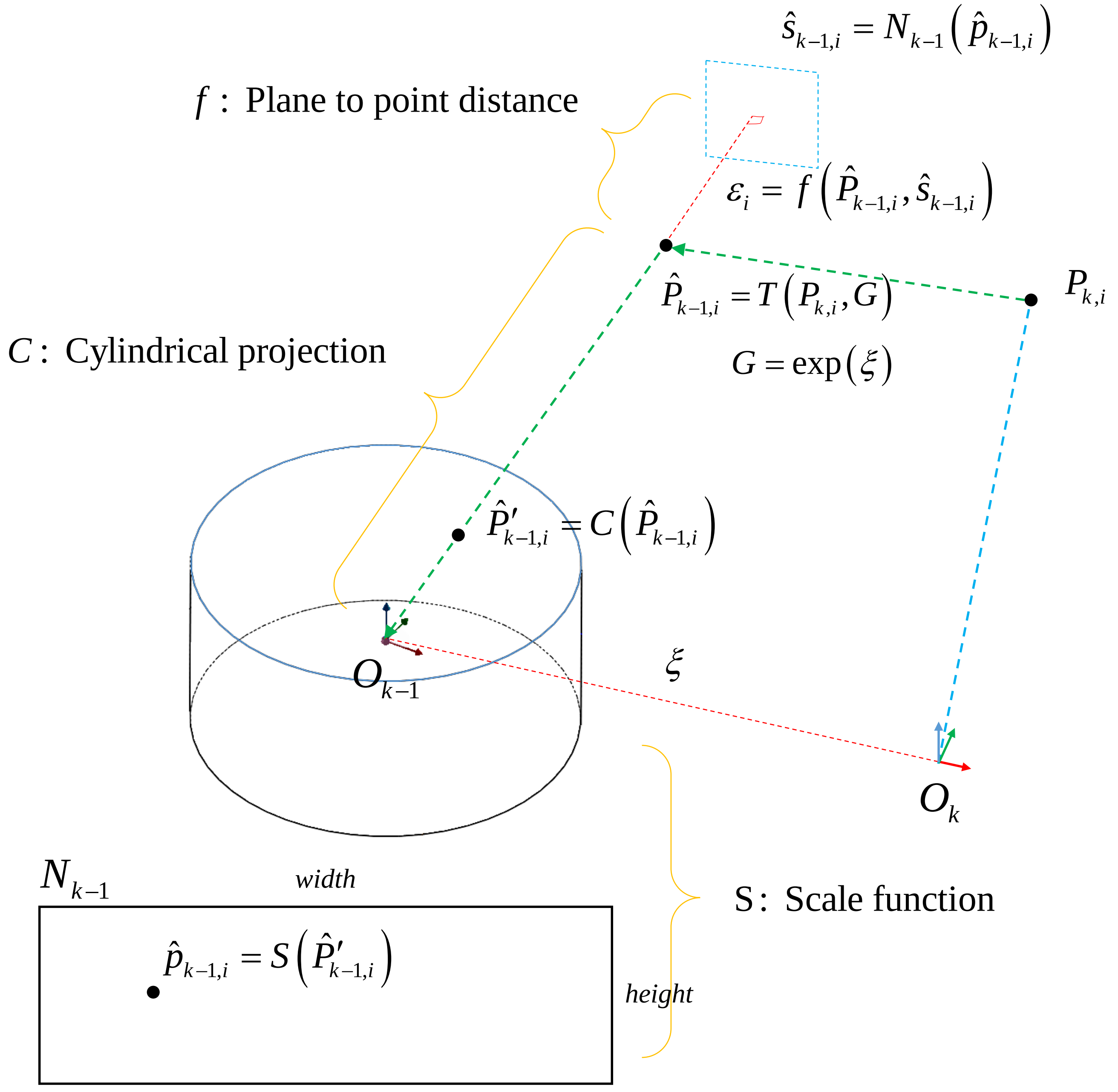


- Proposed method

1. Cylindrical projection
2. Calculate point to plane distance

No search problem!! -> Fast
Differentiable!! -> Numerical optimization

$$\underset{\mathbf{T}}{\operatorname{argmin}}\left(\sum_{i}^{N} f\left(\mathbf{T}\right)^{2}\right)$$

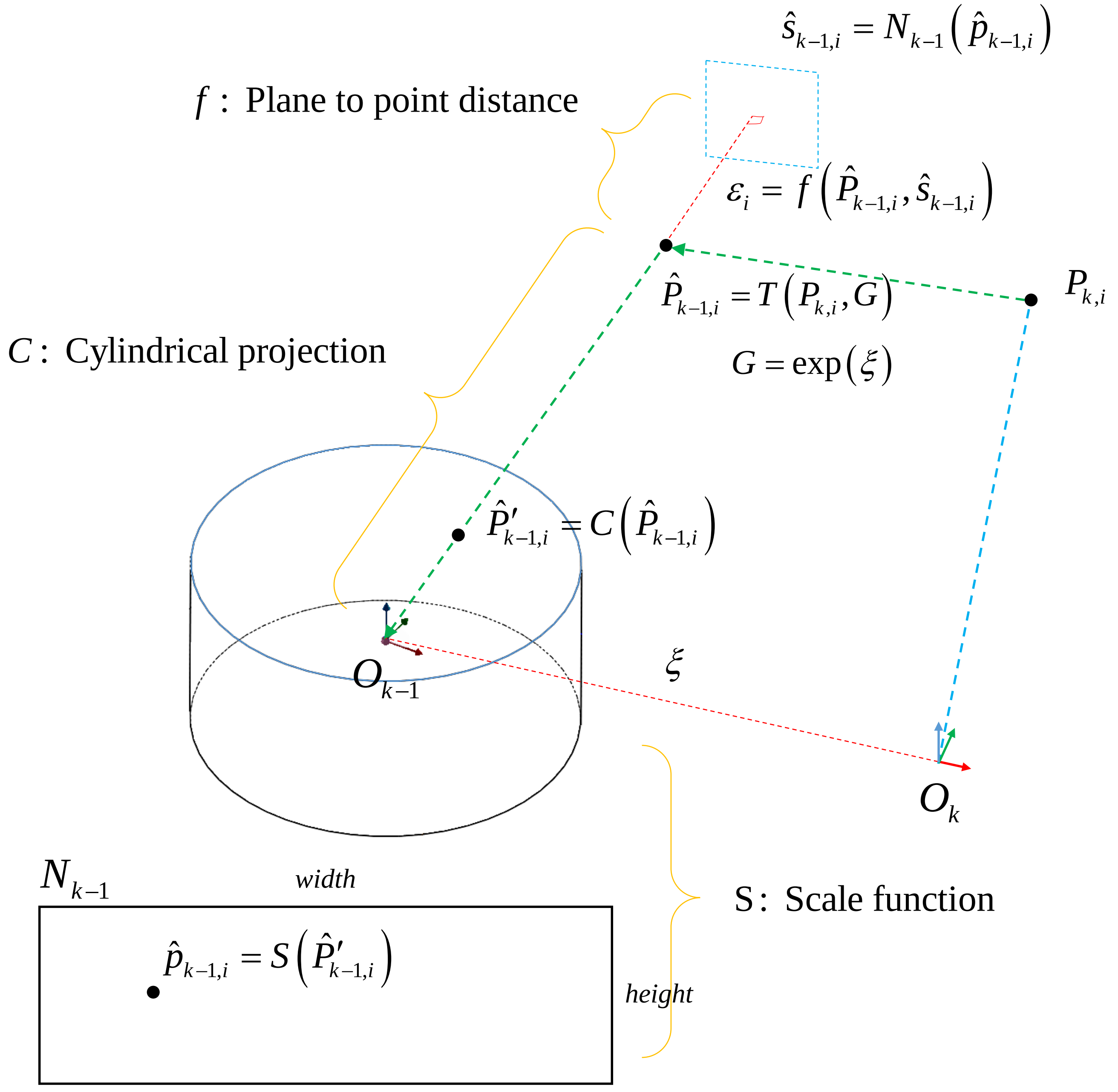


- Rigid body transformation

$$\hat{P}_{k-1,i} = T\left(P_{k,i}, G\right)$$

$$\hat{P}_{k-1,i} = G \cdot P_{k,i}$$

$$G = \exp(\xi)$$



- Cylindrical projection

$$\hat{P}'_{k-1,i} = C\left(\hat{P}_{k-1,i}\right)$$

$$\hat{P}'_{k-1,i} = \begin{bmatrix} x_n & y_n & z_n \end{bmatrix}^T, \ \hat{P}_{k-1,i} = \begin{bmatrix} x & y & z \end{bmatrix}^T$$

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \frac{1}{\sqrt{x^2 + y^2}} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\hat{p}_{k-1,i} = S\left(\hat{P}'_{k-1,i}\right)$$

$$\hat{p}_{k-1,i} = \begin{bmatrix} u & v \end{bmatrix}^T, \ \hat{P}'_{k-1,i} = \begin{bmatrix} x_n & y_n & z_n \end{bmatrix}^T$$

$$\begin{bmatrix} r \\ \theta \\ z \end{bmatrix} = \begin{bmatrix} \sqrt{x_n^2 + y_n^2} \\ \tan^{-1}\left(\frac{y_n}{x_n}\right) \\ z_n \end{bmatrix}, -\pi \le \theta \le \pi, \quad \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} -s_\theta \theta + c_\theta \\ -s_z z + c_z \end{bmatrix}$$

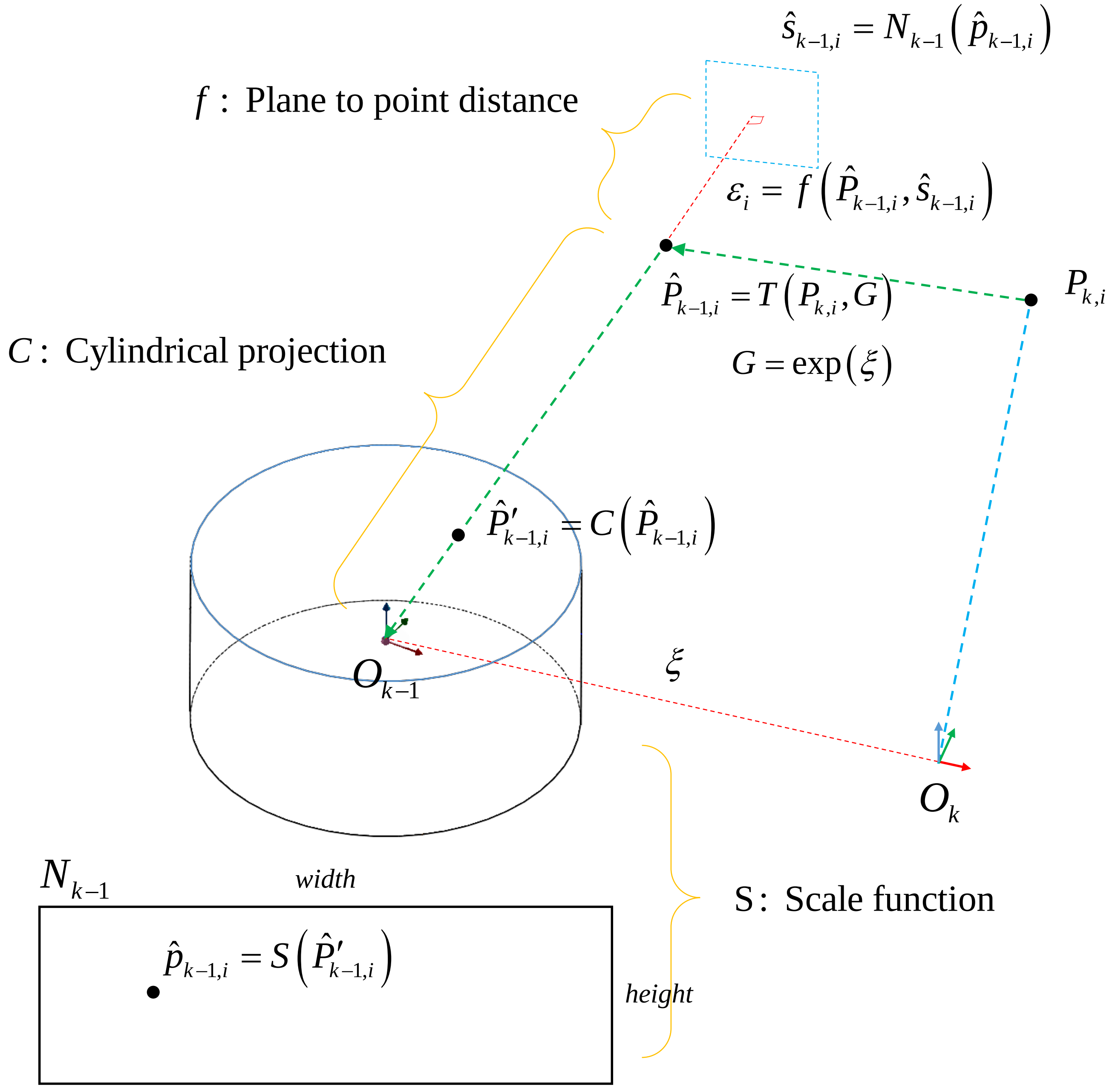
$$s_\theta = width / 2\pi$$

$$c_\theta = width / 2$$

$$s_z = height / \tan(\mathrm{vFov}_{\max} - \mathrm{vFov}_{\min})$$

$$c_z = height \frac{\mathrm{vFov}_{\max}}{\mathrm{vFov}_{\max} - \mathrm{vFov}_{\min}}$$

$$\mathrm{vFov}_{\max} = 2, \ \mathrm{vFov}_{\min} = -24.9$$

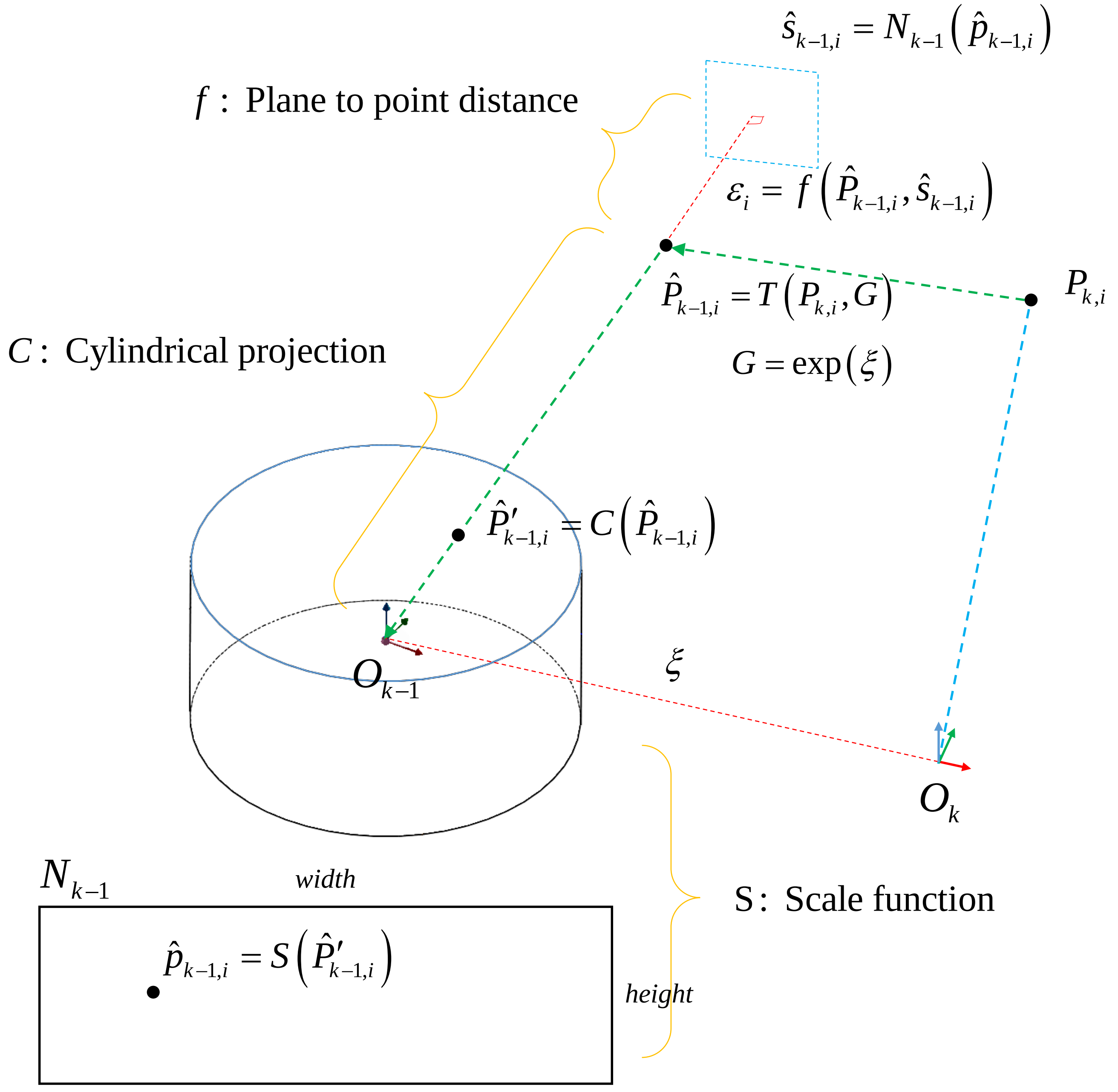


- Plane to point distance

$$\varepsilon_i = f\left(\hat{P}_{k-1,i}, \hat{s}_{k-1,i}\right)$$

$$\hat{P}_{k-1,i} = \begin{bmatrix} x & y & z \end{bmatrix}^T, \hat{s}_{k-1,i} = \begin{bmatrix} a & b & c & d \end{bmatrix}^T$$

$$\varepsilon_i = \frac{|ax + by + cz + d|}{\sqrt{a^2 + b^2 + c^2}}$$



- Cost function

$$E(\xi) = \sum_i \varepsilon_i^{\,2} = \sum_i \left\{ f\left(\hat{P}_{k-1,i}, \hat{s}_{k-1,i}\right)\right\}^2$$

$$\xi^* = \underset{\xi}{\operatorname{argmin}}\left(E(\xi)\right)$$

$$E(\xi) = \sum_i \left\{ f\left(T\left(P_k, \xi\right), N_{k-1}\left(S\left(C\left(T\left(P_k, \xi\right)\right)\right)\right)\right)\right\}^2$$

- Jacobian

$$J = \frac{\partial f}{\partial \xi} = \frac{\partial f}{\partial \hat{P}_{k-1,i}} \frac{\partial \hat{P}_{k-1,i}}{\partial \xi} + \frac{\partial f}{\partial \hat{s}_{k-1,i}} \frac{\partial \hat{s}_{k-1,i}}{\partial \xi}$$

$$= \frac{\partial f}{\partial \hat{P}_{k-1,i}} \frac{\partial \hat{P}_{k-1,i}}{\partial G} \frac{\partial G}{\partial \xi} + \frac{\partial f}{\partial \hat{s}_{k-1,i}} \frac{\partial \hat{s}_{k-1,i}}{\partial \hat{p}_{k-1,i}} \frac{\partial \hat{p}_{k-1,i}}{\partial \hat{P}'_{k-1,i}} \frac{\partial \hat{P}'_{k-1,i}}{\partial \hat{P}_{k-1,i}} \frac{\partial \hat{P}_{k-1,i}}{\partial G} \frac{\partial G}{\partial \xi}$$

$$J=\frac{\partial f}{\partial \xi}=\frac{\partial f}{\partial \hat{P}_{k-1,i}}\frac{\partial \hat{P}_{k-1,i}}{\partial \xi}+\frac{\partial f}{\partial \hat{s}_{k-1,i}}\frac{\partial \hat{s}_{k-1,i}}{\partial \xi}$$

$$=\underbrace{\frac{\partial f}{\partial \hat{P}_{k-1,i}}}_{1}\underbrace{\frac{\partial \hat{P}_{k-1,i}}{\partial G}}_{6}\underbrace{\frac{\partial G}{\partial \xi}}_{7}+\underbrace{\frac{\partial f}{\partial \hat{s}_{k-1,i}}}_{2}\underbrace{\frac{\partial \hat{s}_{k-1,i}}{\partial \hat{p}_{k-1,i}}}_{3}\underbrace{\frac{\partial \hat{p}_{k-1,i}}{\partial \hat{P}'_{k-1,i}}}_{4}\underbrace{\frac{\partial \hat{P}'_{k-1,i}}{\partial \hat{P}_{k-1,i}}}_{5}\underbrace{\frac{\partial \hat{P}_{k-1,i}}{\partial G}}_{6}\underbrace{\frac{\partial G}{\partial \xi}}_{7}$$

1. $$\frac{\partial f}{\partial \hat{P}_{k-1,i}}=\begin{bmatrix}\frac{\partial f}{\partial \hat{P}_{k-1,i,x}} & \frac{\partial f}{\partial \hat{P}_{k-1,i,y}} & \frac{\partial f}{\partial \hat{P}_{k-1,i,z}}\end{bmatrix}$$

2. $$\frac{\partial f}{\partial \hat{s}_{k-1,i}}=\begin{bmatrix}\frac{\partial f}{\partial \hat{s}_{k-1,i,a}} & \frac{\partial f}{\partial \hat{s}_{k-1,i,b}} & \frac{\partial f}{\partial \hat{s}_{k-1,i,c}} & \frac{\partial f}{\partial \hat{s}_{k-1,i,d}}\end{bmatrix}$$

3. $$\frac{\partial \hat{s}_{k-1,i}}{\partial \hat{p}_{k-1,i}}=\begin{bmatrix}\frac{\partial \hat{s}_{k-1,i,a}}{\partial \hat{p}_{k-1,i,u}} & \frac{\partial \hat{s}_{k-1,i,a}}{\partial \hat{p}_{k-1,i,v}}\\ \frac{\partial \hat{s}_{k-1,i,b}}{\partial \hat{p}_{k-1,i,u}} & \frac{\partial \hat{s}_{k-1,i,b}}{\partial \hat{p}_{k-1,i,v}}\\ \frac{\partial \hat{s}_{k-1,i,c}}{\partial \hat{p}_{k-1,i,u}} & \frac{\partial \hat{s}_{k-1,i,c}}{\partial \hat{p}_{k-1,i,v}}\\ \frac{\partial \hat{s}_{k-1,i,d}}{\partial \hat{p}_{k-1,i,u}} & \frac{\partial \hat{s}_{k-1,i,d}}{\partial \hat{p}_{k-1,i,v}}\end{bmatrix}$$

4. $$\frac{\partial \hat{p}_{k-1,i}}{\partial \hat{P}'_{k-1,i}}=\begin{bmatrix}\frac{\partial \hat{p}_{k-1,i,u}}{\partial \hat{P}'_{k-1,i,x}} & \frac{\partial \hat{p}_{k-1,i,u}}{\partial \hat{P}'_{k-1,i,y}} & \frac{\partial \hat{p}_{k-1,i,u}}{\partial \hat{P}'_{k-1,i,z}}\\ \frac{\partial \hat{p}_{k-1,i,v}}{\partial \hat{P}'_{k-1,i,x}} & \frac{\partial \hat{p}_{k-1,i,v}}{\partial \hat{P}'_{k-1,i,y}} & \frac{\partial \hat{p}_{k-1,i,v}}{\partial \hat{P}'_{k-1,i,z}}\end{bmatrix}$$

5. $$\frac{\partial \hat{P}'_{k-1,i}}{\partial \hat{P}_{k-1,i}}=\begin{bmatrix}\frac{\partial \hat{P}'_{k-1,i,x}}{\partial \hat{P}_{k-1,i,x}} & \frac{\partial \hat{P}'_{k-1,i,x}}{\partial \hat{P}_{k-1,i,y}} & \frac{\partial \hat{P}'_{k-1,i,x}}{\partial \hat{P}_{k-1,i,z}}\\ \frac{\partial \hat{P}'_{k-1,i,y}}{\partial \hat{P}_{k-1,i,x}} & \frac{\partial \hat{P}'_{k-1,i,y}}{\partial \hat{P}_{k-1,i,y}} & \frac{\partial \hat{P}'_{k-1,i,y}}{\partial \hat{P}_{k-1,i,z}}\\ \frac{\partial \hat{P}'_{k-1,i,z}}{\partial \hat{P}_{k-1,i,x}} & \frac{\partial \hat{P}'_{k-1,i,z}}{\partial \hat{P}_{k-1,i,y}} & \frac{\partial \hat{P}'_{k-1,i,z}}{\partial \hat{P}_{k-1,i,z}}\end{bmatrix}$$

6. $$\frac{\partial \hat{P}_{k-1,i}}{\partial G}=$$ 3x12

7. $$\frac{\partial G}{\partial \xi}=$$ 12x6

6. $$\frac{\partial \hat{P}_{k-1,i}}{\partial G} = \begin{bmatrix} P_{k,i,x} & 0 & 0 & P_{k,i,y} & 0 & 0 & P_{k,i,z} & 0 & 0 & 1 & 0 & 0 \\ 0 & P_{k,i,x} & 0 & 0 & P_{k,i,y} & 0 & 0 & P_{k,i,z} & 0 & 0 & 1 & 0 \\ 0 & 0 & P_{k,i,x} & 0 & 0 & P_{k,i,y} & 0 & 0 & P_{k,i,z} & 0 & 0 & 1 \end{bmatrix}$$

7. $$\frac{\partial G}{\partial \xi} = \begin{bmatrix} 0 & 0 & 0 & 0 & r_{31} & -r_{21} \\ 0 & 0 & 0 & -r_{31} & 0 & r_{11} \\ 0 & 0 & 0 & r_{21} & -r_{11} & 0 \\ 0 & 0 & 0 & 0 & r_{32} & -r_{22} \\ 0 & 0 & 0 & -r_{32} & 0 & r_{12} \\ 0 & 0 & 0 & r_{22} & -r_{12} & 0 \\ 0 & 0 & 0 & 0 & r_{33} & -r_{23} \\ 0 & 0 & 0 & -r_{33} & 0 & r_{13} \\ 0 & 0 & 0 & r_{23} & -r_{13} & 0 \\ 1 & 0 & 0 & 0 & t_z & -t_y \\ 0 & 1 & 0 & -t_z & 0 & t_x \\ 0 & 0 & 1 & t_y & -t_x & 0 \end{bmatrix}$$

Levenberg-Marquardt method

Algorithm 5: Levenberg-Marquardt algorithm

input : $f : \mathbb{R}^n \to \mathbb{R}$ a function such that $f(\mathbf{x}) = \sum_{i=1}^{m}(f_i(\mathbf{x}))^2$
where all the f_i are differentiable functions from $\mathbb{R}^n$ to $\mathbb{R}$
$\mathbf{x}^{(0)}$ an initial solution

output: $\mathbf{x}^\star$, a local minimum of the cost function f.

```
begin
    k ← 0 ;
    λ ← max diag(JᵀJ) ;
    x ← x⁽⁰⁾ ;
    while STOP-CRIT and (k < k_max) do
        Find δ such that (JᵀJ + λdiag(JᵀJ))δ = Jᵀf ;
        x′ ← x + δ ;
        if f(x′) < f(x) then
            x ← x′ ;
            λ ← λ/ν ;
        else
            λ ← νλ ;
        k ← k + 1 ;
    return x
end
```

[Gradient descent 방법]

$$\mathbf{p}_{k+1}=\mathbf{p}_k-2\lambda_k J_{\mathbf{r}}^T(\mathbf{p}_k)\mathbf{r}(\mathbf{p}_k),\ \ k\geq 0 \quad \text{--- (16)'}$$

[가우스-뉴턴법]

$$\mathbf{p}_{k+1}=\mathbf{p}_k-(J_{\mathbf{r}}^T J_{\mathbf{r}})^{-1}J_{\mathbf{r}}^T\mathbf{r}(\mathbf{p}_k),\ \ k\geq 0 \quad \text{--- (24)'}$$

[Levenberg 방법]

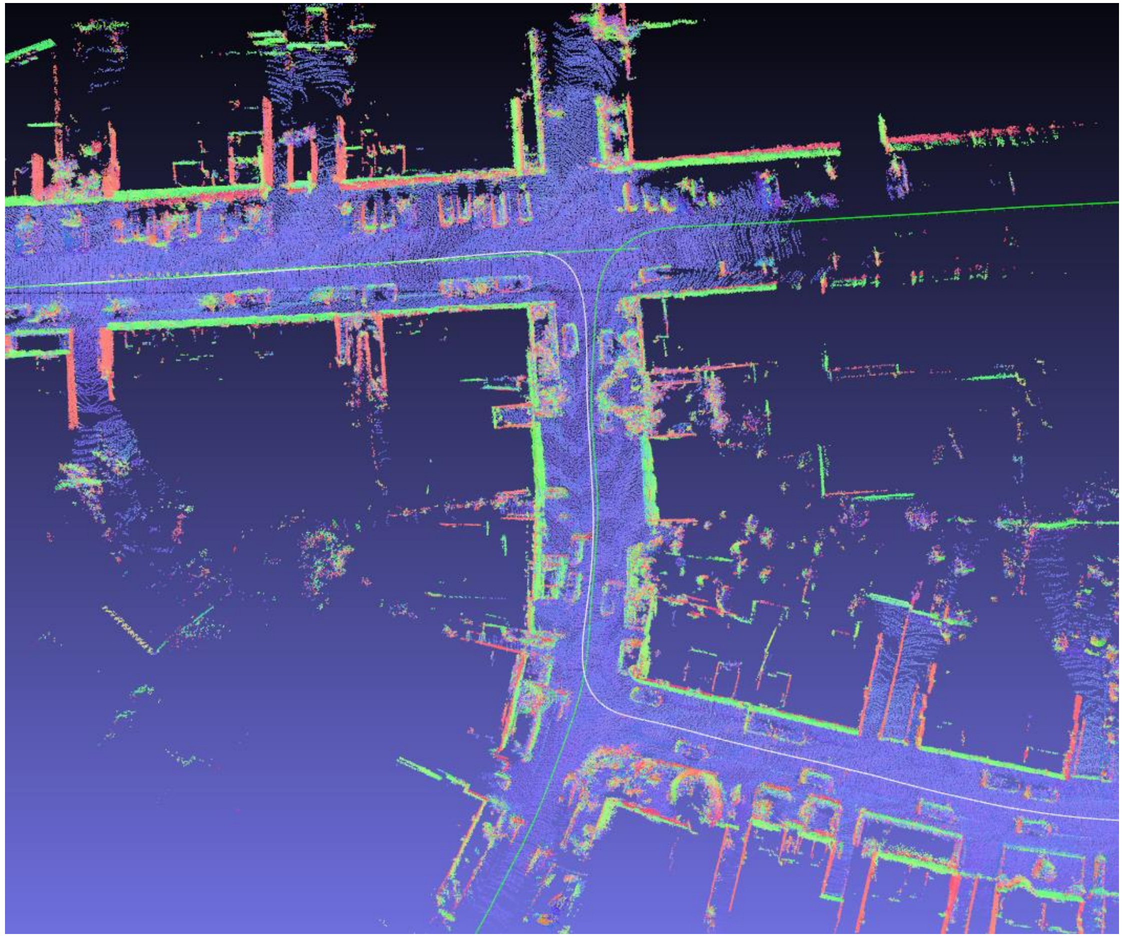
$$\mathbf{p}_{k+1}=\mathbf{p}_k-(J_{\mathbf{r}}^T J_{\mathbf{r}}+\mu_k I)^{-1}J_{\mathbf{r}}^T\mathbf{r}(\mathbf{p}_k),\ \ k\geq 0 \quad \text{--- (30)}$$

[Levenberg-Marquardt 방법]

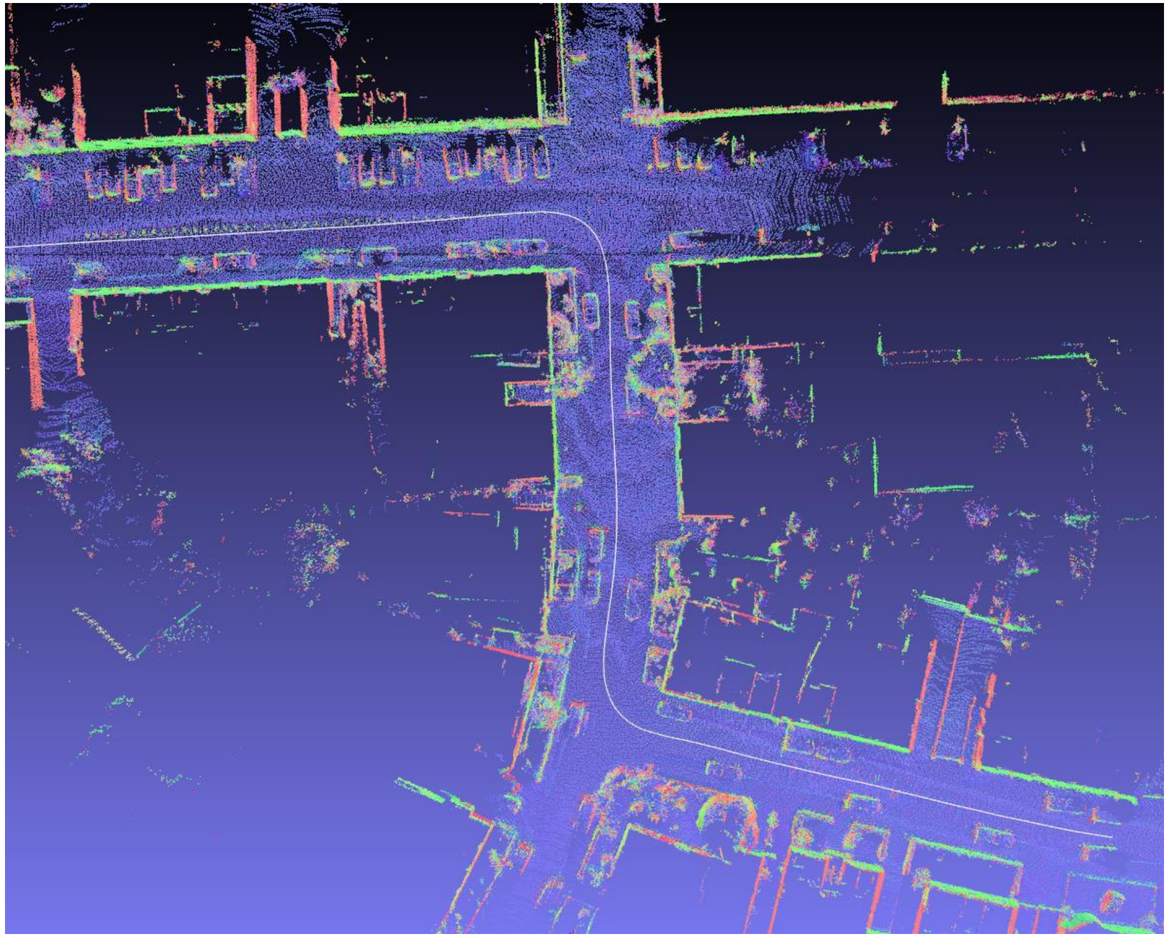
$$\mathbf{p}_{k+1}=\mathbf{p}_k-(J_{\mathbf{r}}^T J_{\mathbf{r}}+\mu_k diag(J_{\mathbf{r}}^T J_{\mathbf{r}}))^{-1}J_{\mathbf{r}}^T\mathbf{r}(\mathbf{p}_k),\ \ k\geq 0 \quad \text{--- (31)}$$

Accuracy

- KITTI Benchmark ground truth



- Proposed method

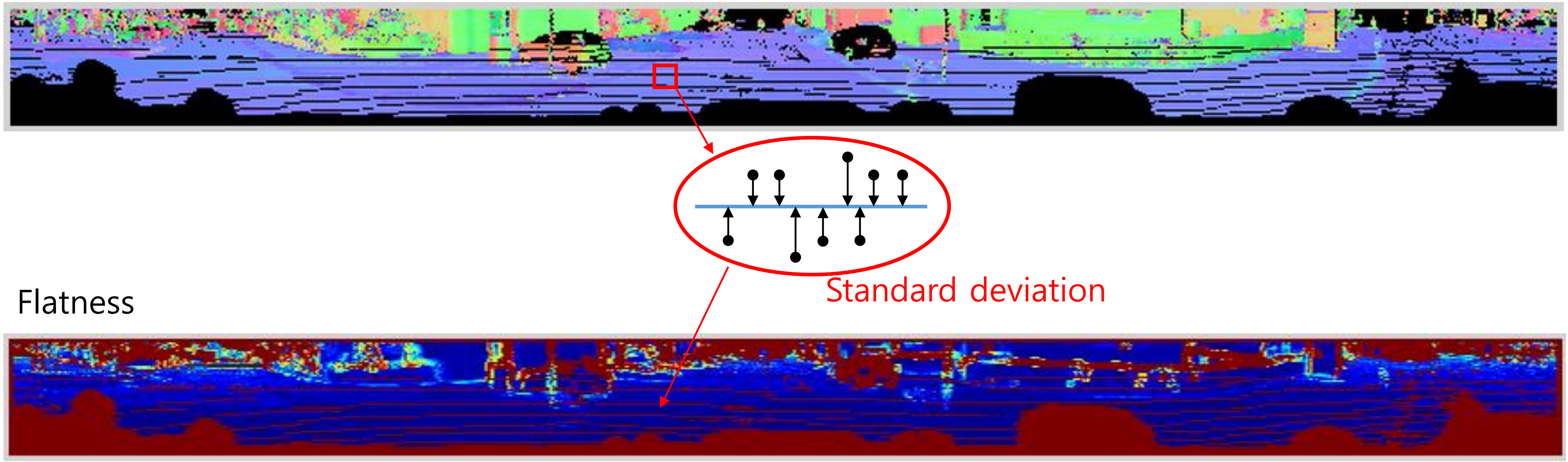


Problem

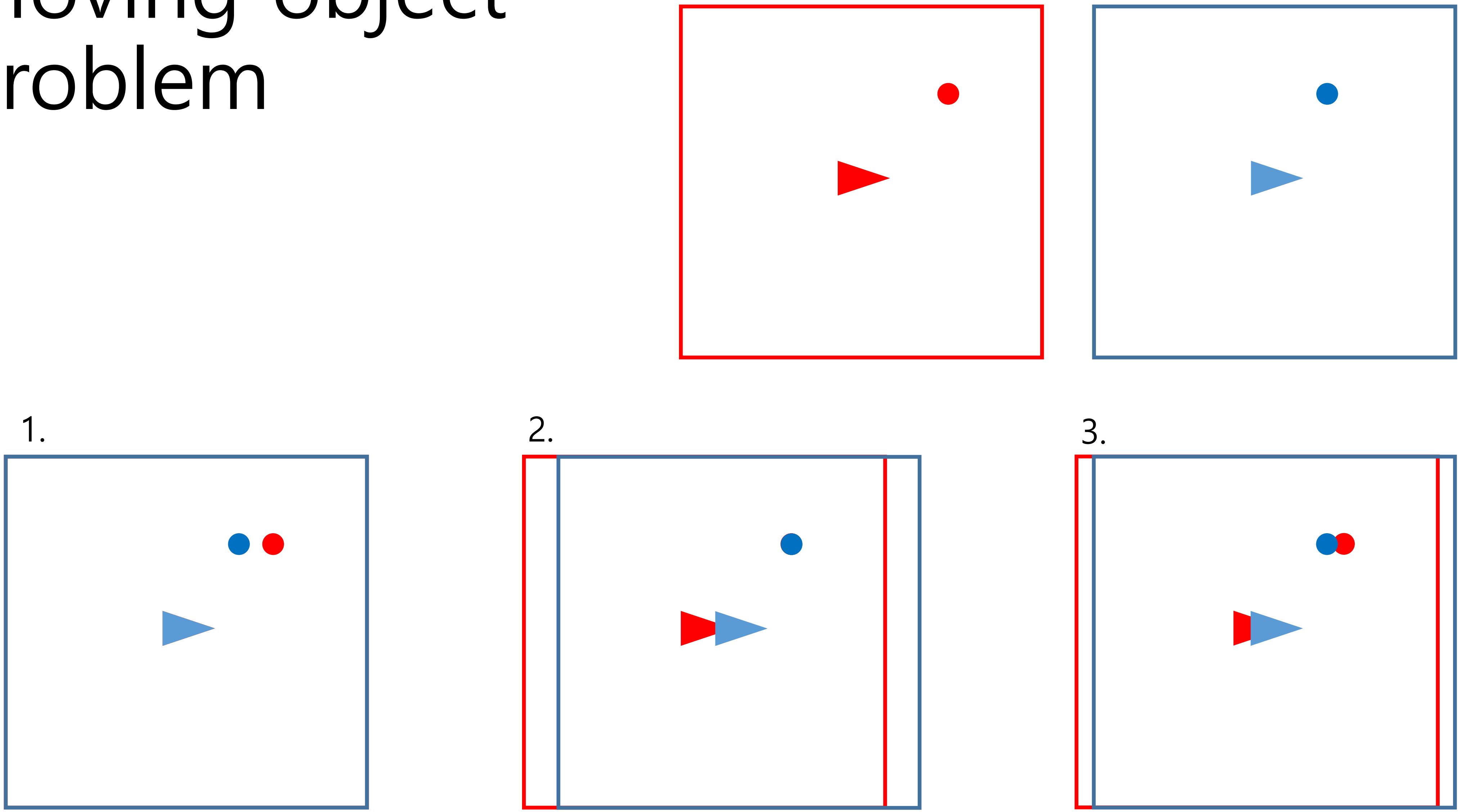
- Non planar area
- Moving object
- Occlusion

Flatness filtering

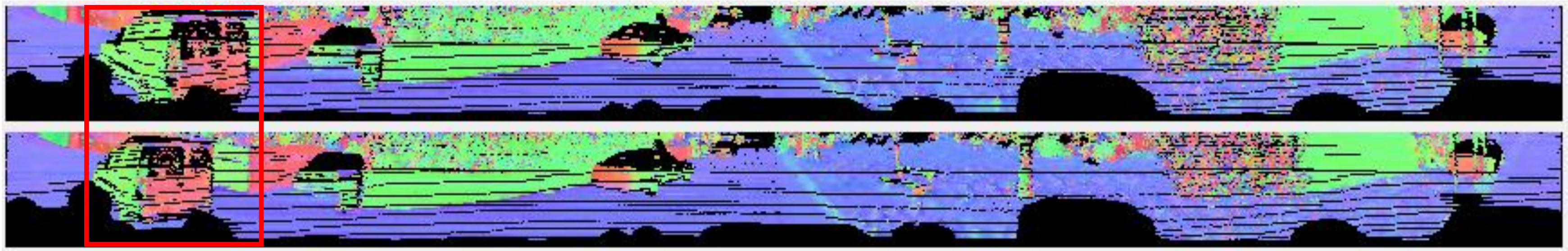
Plane normal

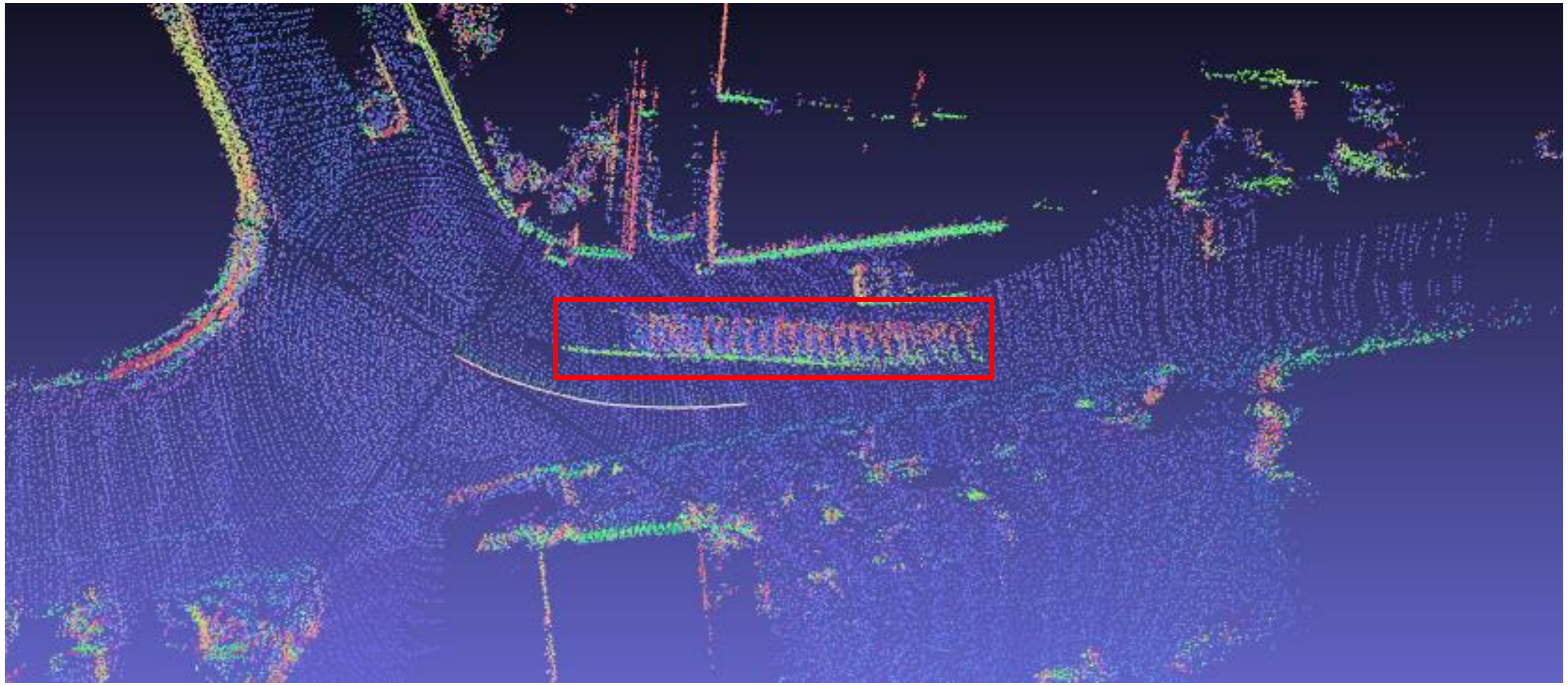


Moving object Problem



Moving object filtering





Conclusion

- Point cloud를 이용한 새로운 pose estimation 알고리즘 제안
- 제안한 알고리즘은 센서 퓨전이나 GPU 가속 없이 빠르고 정확하게 동작
- Moving object removal, loop closing detection등 몇 가지 문제에 대한 해결이 필요하다.