NAVER LABS
Mapping & Localization
Challenge

# PHAROS LOCALIZATION TEAM

기술특징/구현방법

한국과학기술원 건설 및 환경공학과

석사과정 이상민

한국과학기술원 로봇공학학제전공

석사과정 이중구

KAIST | IRiS Interactive Robotic Systems Lab

# 1. Localization Process

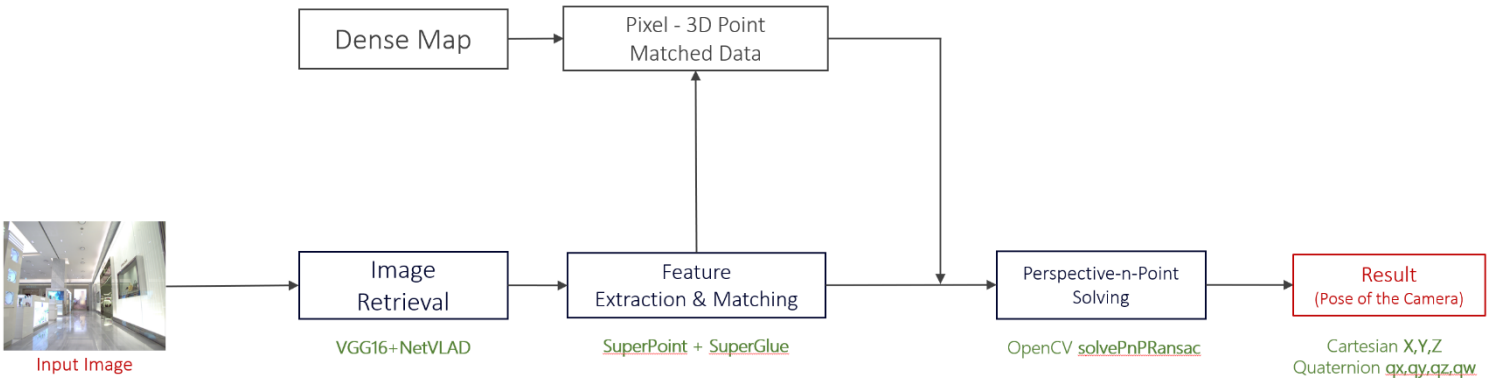


**Figure 1. Indoor Localization Pipeline**

사진 한장으로 실내에서 정확한 위치와 자세를 알 수 있는 저희 팀의 Visual Localization 기술의 경우 다음과 같은 Process를 거치게 됩니다.

A. Input 이미지 입력

B. Pretrained NetVLAD를 통해 1층, 지하 1층 Database에서 Image Retrieval

C. Top 5 NetVLAD Output에서 가장 많은 Inlier Feature Matching 결과를 반환하는 사진에 대한 Feature Extraction 및 Matching 실시

D. LiDAR Raw데이터를 쌓아 만든 Dense Map에 기반하여 Query Image의 Matched Feature, Database Image의 Matched Feature, Database Image위치에서의 Local Map을 이용, Perspective-n-Point와 RANSAC알고리즘을 이용하여 Query Image의 정확한 위치와 자세를 도출

이를 통해 사진한장으로 실내에서 정확한 위치와 자세를 알 수 있게 됩니다. 저희 팀은 크게 3가지 사항을 중심으로 대회를 진행했습니다. 첫번째로, Image Retrieval Accuracy, 두번째로 Matched Feature Accuracy, 세번째로 Map Accuracy입니다.

# 2. Methodology Detail

**A. NetVLAD:** Image Retrieval Accuracy

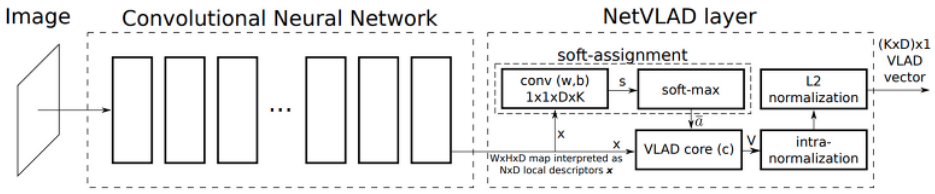


**Figure 2. NetVLAD network, FC layer change to NetVLAD Layer**

- NetVLAD는 기존 VLAD기반 Descriptor와 달리 Neural Network를 이용하여 VLAD에 Flexibility를 추가하였고, 결과적으로 기존 State of art보다 뛰어난 성능을 발휘하는 Image Retrieval 시스템입니다. 기존 ORB-SLAM에서 이용되었던 Bag of Visual Words시스템인 DBoW와 달리, VLAD는 훈련된 각 Cluster center와의 거리 정보를 포함하고 있으며, 이를 통해 더 정확한 Image retrieval이 가능합니다. NetVLAD는 여기서 발전하여 Neural Network를 이용하여 VLAD Decoupling을 실시, 기존 VLAD보다 더 나은 성능을 기대할 수 있는 Flexibility를 부여하였습니다.

$$V(j,k) = \sum_{i=1}^{N} a_k(\mathbf{x}_i)(x_i(j) - c_k(j)) \qquad V(k) = \sum_{i=1}^{N} \frac{e^{w_k^T \mathbf{x}_i + b_k}}{\sum_{k'} e^{w_{k'}^T \mathbf{x}_i + b_{k'}}} (x_i - c_k)$$

**Equation 1. Original VLAD descriptor (LEFT) and NetVLAD Descriptor (RIGHT)**

- *Pipeline*에서는 제공된 pretrained Network를 사용하는 것이 아닌, 저자가 공개한 Matlab MatConvNet Toolbox기반 코드를 통해 Indoor Dataset를 직접 훈련시켰습니다.

- 네트워크 훈련 파라미터는 이와 같습니다.

| | | | |
|---|---|---|---|
| whichSet | 'train' | whichSet | 'val' |
| dbImageFns | *11920x1 cell* | dbImageFns | *11917x1 cell* |
| utmDb | *2x11920 double* | utmDb | *2x11917 double* |
| qImageFns | *1704x1 cell* | qImageFns | *1704x1 cell* |
| utmQ | *2x1704 double* | utmQ | *2x1704 double* |
| numImages | 11920 | numImages | 11917 |
| numQueries | 1704 | numQueries | 1x1 double |
| posDistThr | 2 | posDistThr | 2 |
| posDistSqThr | 4 | posDistSqThr | 4 |
| nonTrivPosDistSqThr | 8 | nonTrivPosDistSqThr | 8 |

**Figure 3-(a). 1F NetVLAD Train Parameter**

| | | | |
|---|---|---|---|
| whichSet | 'train' | whichSet | 'val' |
| dbImageFns | *11721x1 cell* | dbImageFns | *11715x1 cell* |
| utmDb | *2x11721 double* | utmDb | *2x11715 double* |
| qImageFns | *1676x1 cell* | qImageFns | *1675x1 cell* |
| utmQ | *2x1676 double* | utmQ | *2x1675 double* |
| numImages | 11721 | numImages | 11715 |
| numQueries | 1676 | numQueries | 1675 |
| posDistThr | 1.5000 | posDistThr | 1.5000 |
| posDistSqThr | 2.2500 | posDistSqThr | 2.2500 |
| nonTrivPosDistSqThr | 6 | nonTrivPosDistSqThr | 6 |

**Figure 3-(b). B1 NetVLAD Train Parameter**

- 지하 1층의 경우 많은 Noise들이 존재하여, Distance Threshold를 더 낮춰서 훈련시켰습니다.

**B. SuperPoint + SuperGlue:** Matched Feature Accuracy

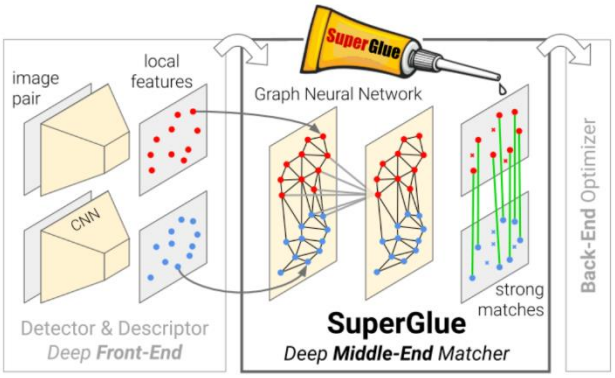


**Figure 4. SuperPoint + SuperGlue Process**

- *SuperPoint*는 CNN기반 Feature Detector로, Self-Supervised Approach를 통해 네트워크를 훈련합니다. Synthetic Shape을 통해 이미지에서 Feature특성을 가질 수 있는 부분을 훈련 후, 이를 feature point label이 없는 데이터셋에 적용시킵니다. 계산된 Magic Point를 이용하여 Homography계산을 통해 최종적으로 Interest Point를 계산합니다.

- *SuperGlue*는 Feature Matching Cost를 Deep Neural Network로 풀어냈습니다. 먼저 계산된 Feature와 Feature Confidence를 이용하여 Attentional Graph Neural Network를 통해 각 Key point에 대하여 반복적으로 Feature사이 관계를 풀어냅니다. 그 후 Optimal Matching Layer를 통해 계산된 Score Matrix를 이용하여 Best Matched Feature를 계산합니다.

- *Pipeline*에서는 1층과 지하1층에 대하여 다른 parameter를 적용하여 최대한 성능을 최적화시켰습니다. 1층의 경우, Matching Confidence가 0.6이상인 Feature Point만 사용하였으며, 지하 1층의 경우 적은 조도로 인한 Motion Blur와 Image Noise를 고려하여 Matching Confidence가 0.65이상인 경우만 사용하였습니다. 또한 이미지마다 다른 밝기, 대비, 해상도를 가지고 있으므로, 일관된 Feature 추출과 Matching성능 확보를 위해 Query, Database 이미지에 대하여 Resize와 Histogram Equalization을 적용했습니다.

| Floor | Algorithm | Non-Maximum Suppression Radius | Match Confidence | Image Size (width) |
|---|---|---|---|---|
| 1F | SuperPoint | **5** | - | **600** |
| | SuperGlue | - | **>0.6** | **600** |
| B1 | SuperPoint | **5** | - | **600** |
| | SuperGlue | - | **>0.65** | **600** |

**Table1. Difference between 1F and B1 Parameter**

- SuperGlue의 경우 Indoor weight를 먼저 사용했으나, pre-trained network의 indoor weight는 백화점과 같은 광범위한 Feature가 아닌 일반적인 사무환경에서 train이 되었음을 파악하여, 광량 변화와 더 많은 Feature matching에 대응할 수 있는 Mega-Depth dataset으로 training된 outdoor weight를 사용했습니다.

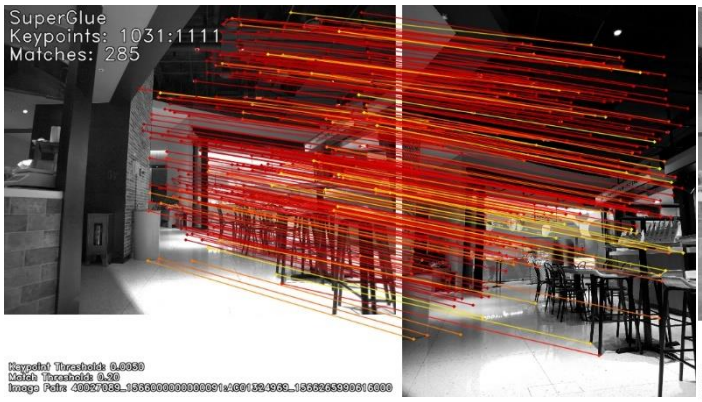


Figure 5-(a). Original (Matched Feature: 285EA)

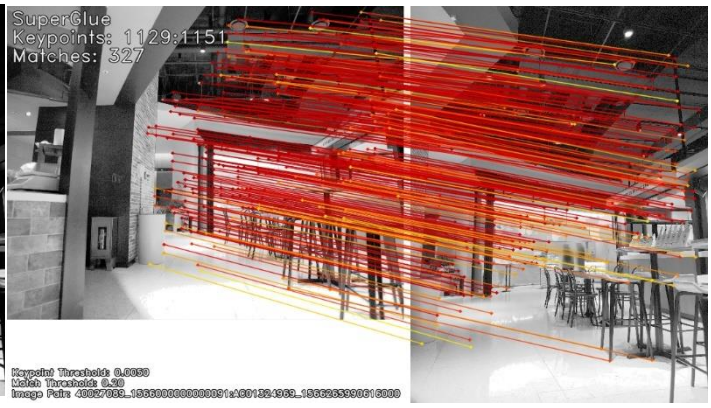


Figure 5-(b). Histogram Eq. (Matched Feature: 327EA)


Figure 6-(a). Indoor Weight (Matched Feature: 254EA)


Figure 6-(b). Outdoor Weight (Matched Feature: 332EA)
(Higher Matching Confidence with More Features)

## C. **Dense Map:** Map Accuracy

- 평면으로 표현되는 사진과 달리 PointCloud는 3D 좌표계 전체에 대한 정보를 가지고 있습니다. 따라서 저희는 이미지 정보와 PointCloud 정보를 결합하여 이미지의 각 픽셀이 3D 좌표계에서 어떤 위치에 대응되는지를 찾고자 하였습니다.

- fig. 7-(a) 같이 Sparse Map을 사용하면 물체의 표면에 대응되는 Point의 개수가 이미지 픽셀을 모두 채울 정도로 충분하지 않아 실제 물체의 뒤에 있는 data에 잘못 대응되는 경우가 많았습니다. 이러한 문제를 해결하고자 Raw LiDAR Data를 쌓은 후 Point를 1cm간격으로 Down sampling하여 fig. 7-(b)와 같은 Dense Map을 생성하여 mismatched outlier를 제거하였습니다.

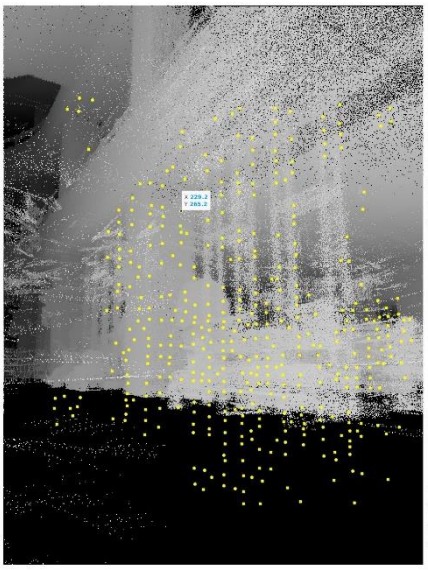

Figure 7-(a). Sparse Map Feature Matched

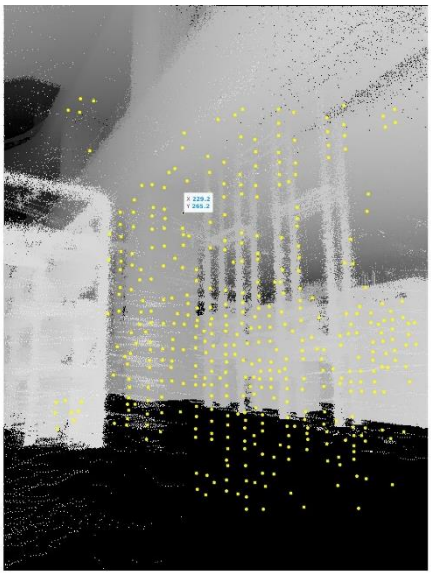

Figure 7-(b). Dense Map Feature Matched

Figure 7-(c). Original Image

- 이미지 각 픽셀의 3D 좌표계에서의 위치를 구하기 위해서는 PointCloud의 점들을 모두 Image Plane에 투영시키는 방법을 사용하였습니다.

$$s \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \cdot [R \mid t] \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- 추가적으로, 이미지 해상도가 Point의 해상도보다 월등히 커 대부분의 픽셀에 3D 좌표가 대응되지 않는 문제점을 해결하기 위해 이미지의 해상도를 1/4로 down sampling한 후 각 X,Y,Z 값을 대입하였습니다.

### C-2. People Extraction

- 사람의 움직임으로 인해 Raw Point Cloud 데이터에는 많은 노이즈가 있습니다. 이러한 노이즈를 제거하기 위해서는 로봇이 지나간 Path를 기준으로 일정 거리 내의 Point를 모두 제거하는 방법을 사용하였습니다.
- 로봇이 움직인 Path를 모두 모아 0.5m 간격으로 down sample한 후 각 Path 마다 앞뒤 1.5m, 좌우 1.5m, 높이 1.85m 안의 Point를 모두 제거해 노이즈 문제를 해결했습니다.


**Figure 8. Point Cloud Submap with Noise (People)**

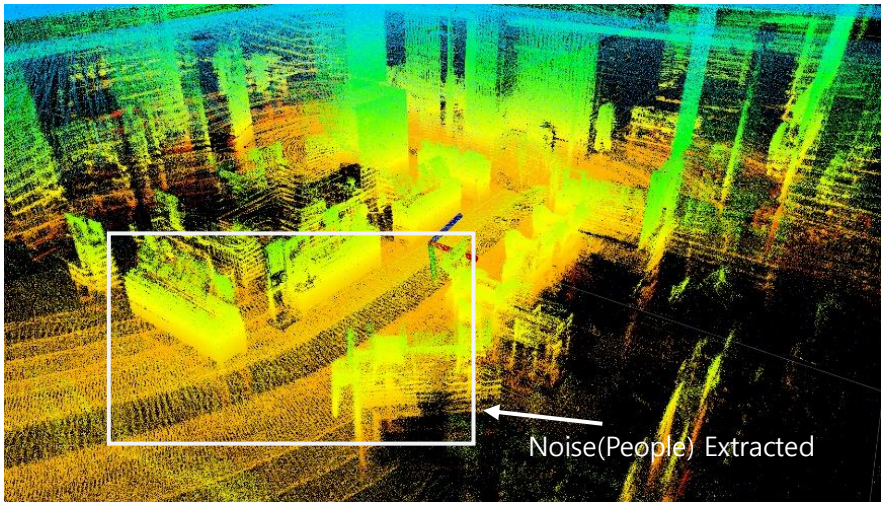


**Figure 9. Point Cloud Submap without Noise (People)**

### C-3. Floor Extraction

- SuperPoint와 SuperGlue를 이용한 특징점 생성 및 매칭 과정에서 백화점 바닥이 대리석 재질이라 반사가 잘 되어 바닥의 특징점들이 매칭되는 경우가 많았습니다. 이러한 mismatch들을 제거하기 위해서 바닥의 Point들을 모두 제거했습니다.


**Figure 10. Floor Point Mismatched**

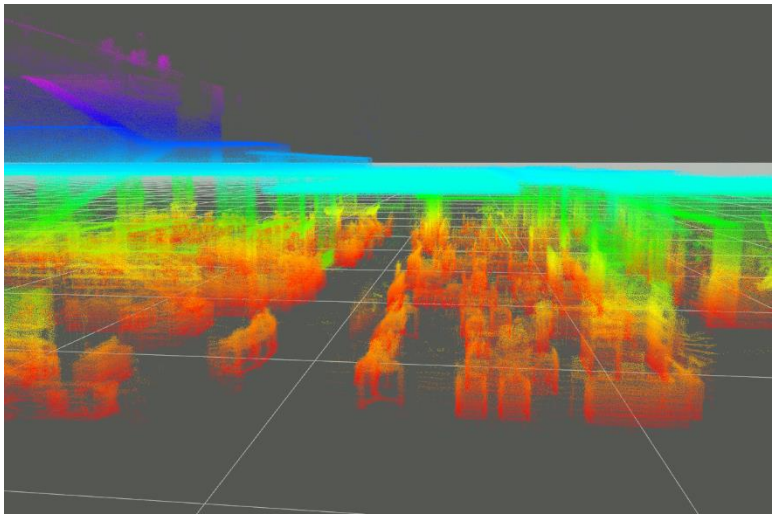

**Figure 11. Point Cloud with Floor Extraction**