

# GNN (Graph Neural Network) 상세 설계

이 문서는 overall\_approach.md의 Stage 5-6 (Graph Construction + GNN Enhancement)의 상세 내용을 다룹니다.

## 1. 개요

### 1.1 목적

독립적인 spectral histogram에 **trajectory context**를 주입하여 discriminative power를 향상시킵니다. 3-layer Graph Attention Network (GAT)를 통해 temporal neighbors의 정보를 message passing으로 전파합니다.

### 1.2 설계 원칙

원칙	설명
Dimension Preservation	입력 800D → 출력 800D (동일 차원)
Residual Learning	입력 histogram 정보 보존
Motion-aware Attention	거리 + 회전 기반 edge feature로 trajectory 패턴 학습

## 2. Hybrid Graph Construction (Temporal + Spatial)

### 2.1 그래프 구조

각 keyframe이 하나의 노드이며, 두 종류의 edge로 연결됩니다:

$$V = \{v_1, v_2, \dots, v_N\}, \quad x_i \in \mathbb{R}^{800}$$

**Temporal edges (항상):**

$$\mathcal{N}_{temp}(i) = \{j : |i - j| \leq k\}$$

**Loop closure edges (검증 후):**

$$\mathcal{N}_{loop}(i) = \{j : (i, j) \in \text{VerifiedLoops}\}$$

**전체 이웃:**

$$\mathcal{N}(i) = \mathcal{N}_{temp}(i) \cup \mathcal{N}_{loop}(i)$$

Edge Type	조건	역할
Temporal	시간적 인접	Trajectory context
Loop Closure	기하 검증 통과	Spatial context

## 2.2 Edge Features

각 edge에 2D feature 벡터 부여 (거리 + 회전):

Feature	계산	정규화
Distance	$d_{ij} = \ t_i - t_j\ _2$	$\log(1 + d)/5$
Rotation	$\theta_{ij} = \arccos\left(\frac{\text{trace}(R_j R_i^T) - 1}{2}\right)$	$\theta/\pi$

$$\mathbf{e}_{ij} = [\tilde{d}_{ij}, \tilde{\theta}_{ij}] \in \mathbb{R}^2$$

**효과:** 이동 거리뿐 아니라 회전 패턴도 attention에 반영하여 trajectory 특성 학습

## 3. GNN Architecture

### 3.1 구조

Input (N, 800)  $\rightarrow$  [GAT Layer  $\times$  3]  $\rightarrow$  Output (N, 800)

각 layer: GATConv  $\rightarrow$  BatchNorm  $\rightarrow$  ReLU  $\rightarrow$  Dropout (+ Residual)

### 3.2 GAT Attention Mechanism

**Edge feature 반영 attention:**

$$e_{ij} = \text{LeakyReLU}(\mathbf{a}^T [W h_i \| W h_j \| \mathbf{e}_{ij}])$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})}$$

**Message aggregation:**

$$h'_i = \sigma \left( \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \cdot W h_j \right)$$

Edge feature  $\mathbf{e}_{ij}$ 가 attention 계산에 직접 반영됨

### 3.3 Residual Connections

- 중간 layer:  $h^{(l+1)} = h^{(l+1)} + h^{(l)}$
- 최종 출력:  $z = h^{(L)} + x$  (input residual)

**효과:** Raw histogram 정보 보존, gradient flow 개선, over-smoothing 방지

## 4. 하이퍼파라미터

파라미터	기본값	설명
Input/Hidden/Output dim	800	Per-elevation histogram
Number of layers	3	GAT layer 수 (임의 설정)
Temporal neighbors	2	이전 2 + 이후 2 프레임
Edge dim	2	Distance + Rotation
Dropout rate	0.1	Regularization

**Note:** Layer 수(3)는 임의로 설정된 값이며, 추후 튜닝 대상입니다.

## 5. Training

### 5.1 Triplet Loss

$$\mathcal{L} = \sum_{(a,p,n)} \max(0, \|z_a - z_p\|_2 - \|z_a - z_n\|_2 + m)$$

조건	정의
Positive pairs	거리 < 5m, temporal gap > 30 frames
Negative pairs	거리 > 30m
Margin	0.1

## 5.2 Per-Sequence Mining

서로 다른 sequence 간 triplet 생성을 방지하여 false negative 발생을 막습니다.

## 6. Message Passing 분석

### 6.1 정보 전파

Layer	도달 범위
1	직접 이웃 (4개)
2	2-hop 이웃 (~8개)
3	3-hop 이웃 (~12개)

### 6.2 Over-smoothing 방지

방법	효과
Residual connection	Input 정보 보존
BatchNorm	Feature scale 유지
Limited depth (3 layers)	과도한 smoothing 방지

## 7. 실험 결과

Dataset	Without GNN	With GNN	변화
HeLiPR (train)	98.5%	100%	+1.5%
KITTI 00	60.14%	62.79%	+2.65%

### Edge Feature 중요성

Edge Feature	설명
None	Uniform attention
Distance only	거리 기반 attention
<b>Distance + Rotation</b>	Trajectory 패턴 학습

**기대 효과:** 회전이 많은 trajectory vs 직진 trajectory 구분 가능

## 8. 요약

구성요소	역할
<b>Temporal Graph</b>	Keyframe 간 시간적 연결 표현
<b>GAT Layer</b>	Attention 기반 message passing
<b>Edge Feature</b>	거리 + 회전으로 trajectory 패턴 학습
<b>Residual</b>	Input histogram 정보 보존
<b>Triplet Loss</b>	Metric learning으로 discriminative embedding

문서 생성일: 2026-01-28