

# 3D Point Clouds

## Lecture 8 – Feature Description

主讲人 黎嘉信

Aptiv 自动驾驶  
新加坡国立大学 博士  
清华大学 本科

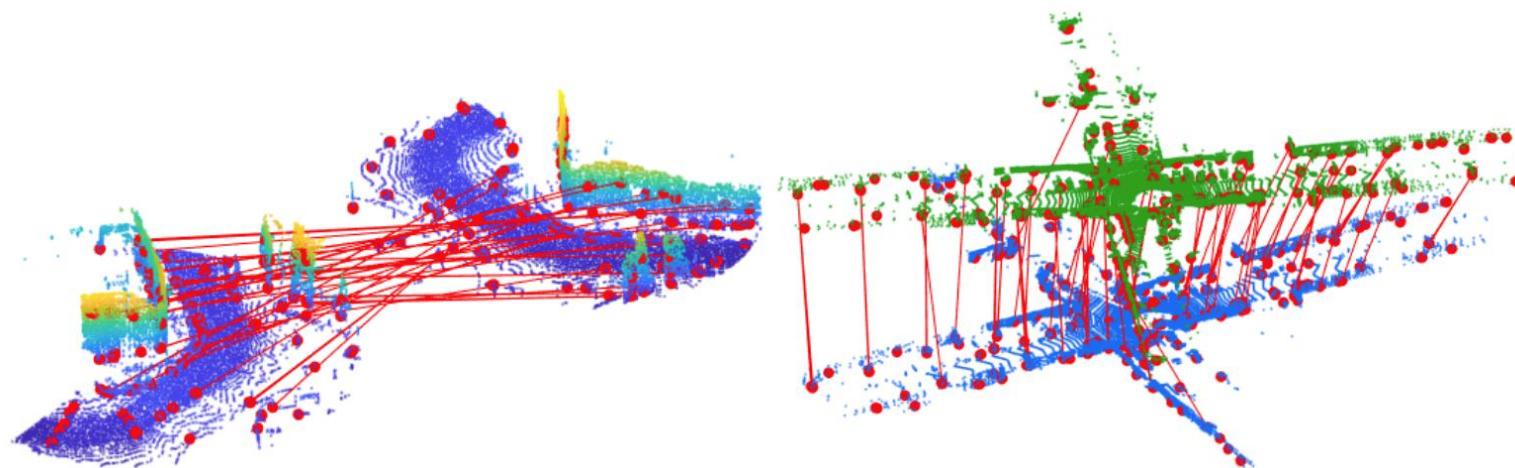


- 1. Introduction
- 2. Classical – PFH, FPFH, SHOT
- 3. Modern – Deep Learning Methods



## Features

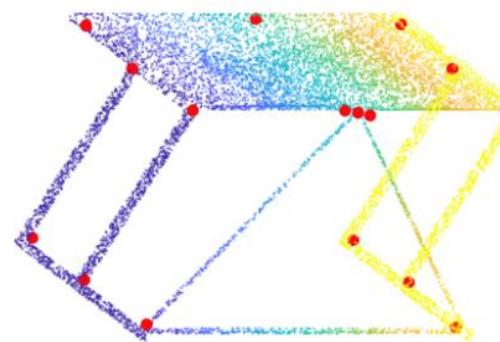
- **Detection:** Identity the **keypoints**
- **Description:** Extract a vector around the keypoint to descript it.
- **Matching:** Determine correspondence between descriptors



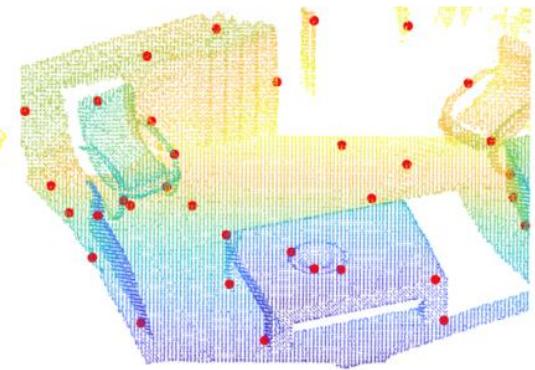


- Handcrafted

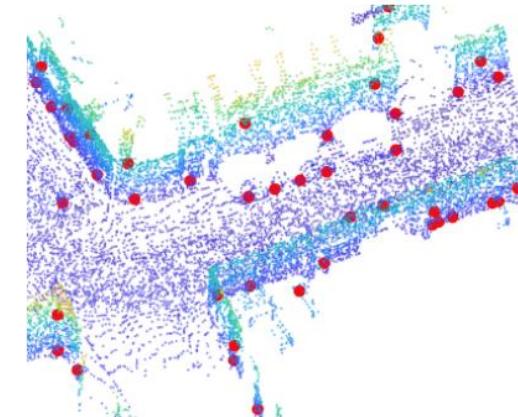
- Harris family
  - Harris 3D with/without intensity
  - Harris 6D with intensity
- ISS



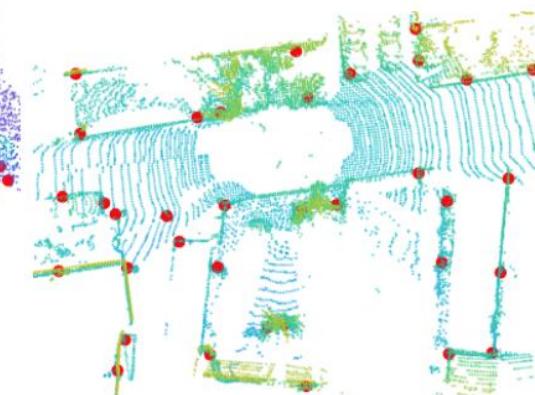
CAD Model



RGBD



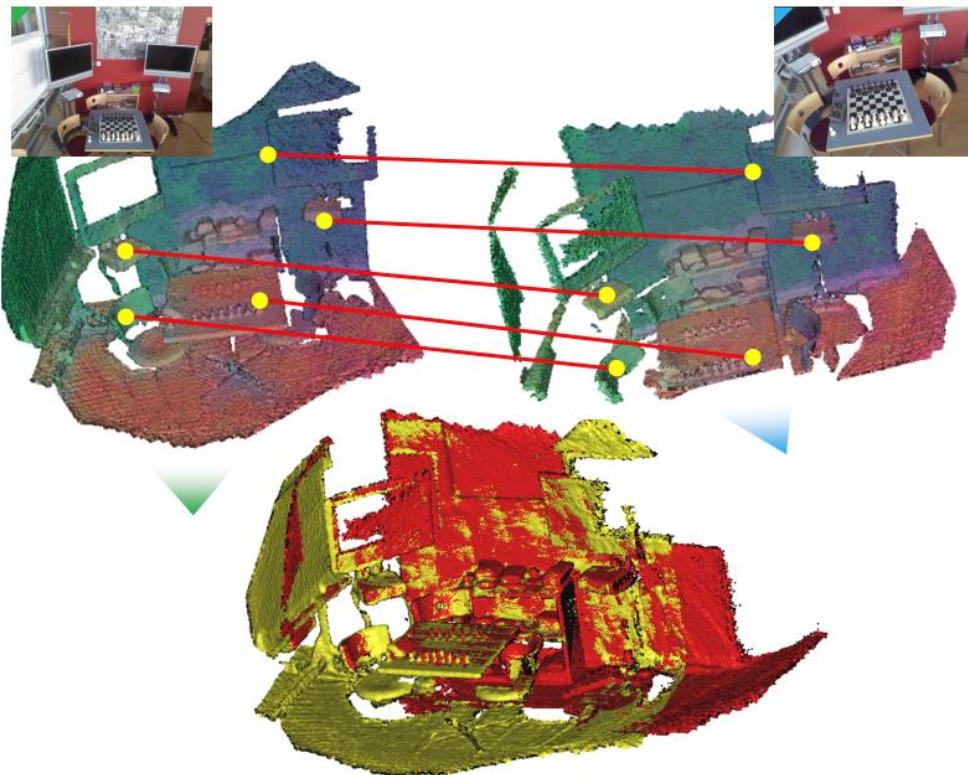
Oxford Robotcar



KITTI

- Deep learning

- USIP

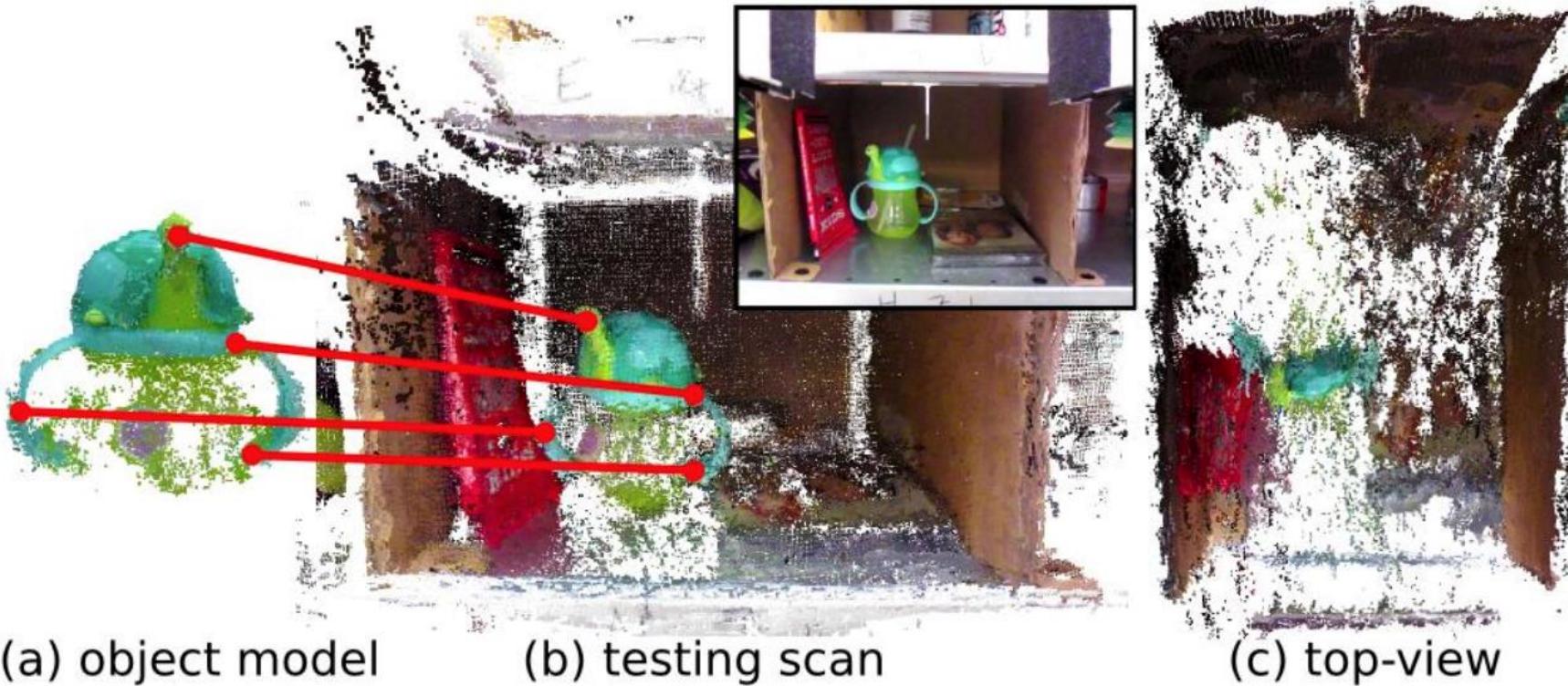


## Registration

- Find a transform to align two point clouds
- A transform consists of
  - Rotation  $R$
  - Translation  $t$
- Method 1 - Iterative Closest Point (ICP)?
  - ICP requires proper initial guess
  - Low overlapping ratio
- Method 2 – Detect and match features
  - No initialization required
  - Works for low overlapping ratio



## Point Cloud Features – Object 6D Pose



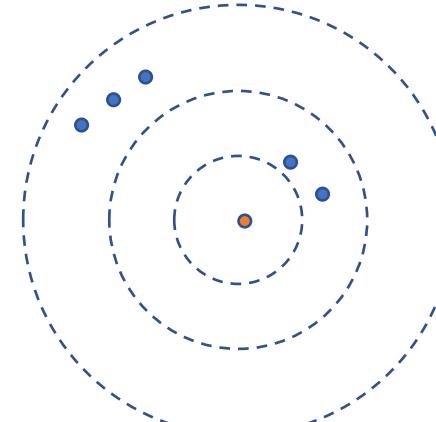
- In fact, it is still registration

Source: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions

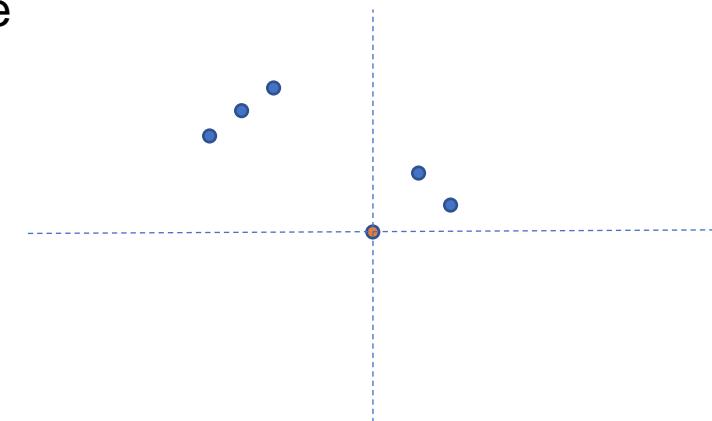


## Point Cloud Feature Descriptor - Handcrafted

- Histogram based
  - Encode local geometric variations, put them into a histogram
  - Methods:
    - Point Feature Histogram (PFH)
    - Fast Point Feature Histogram (FPFH)
- Signature based
  - Compute geometric measures based on local reference frame
  - Methods
    - Structure Indexing
    - Signature of Histograms of OrientTations (SHOT)



0	2	3
---	---	---



Avg(Quadrant 1)	Avg(Quadrant 2)	Avg(Quadrant 3)	Avg(Quadrant 4)
-----------------	-----------------	-----------------	-----------------

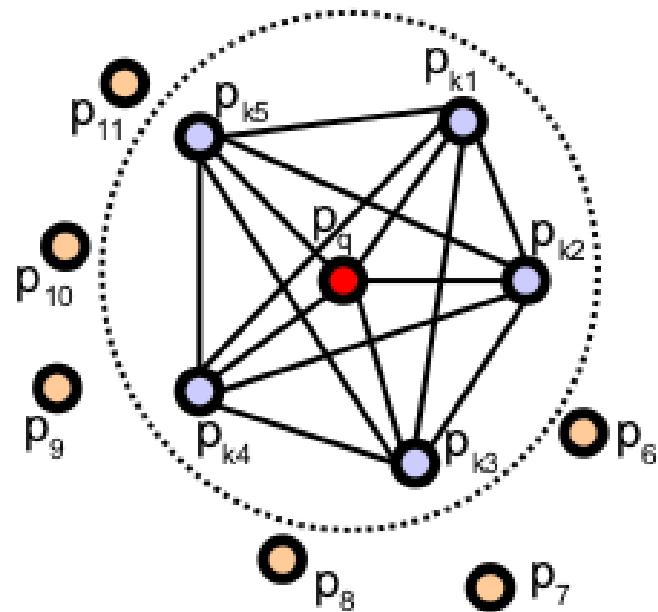


- Input
  - Point coordinates of the point cloud
  - Surface normal vectors of the point cloud
  - A query point
- Output
  - Array of length  $B^3$ 
    - $B$  is the number of bins on each histogram, e.g.,  $B = 5$
    - 3 - there are 3 histograms
- Ideas
  - 6D-Pose independent
  - Captures surface variations in a neighborhood



## PFH – Neighborhood

- Captures surface variations in a **neighborhood**
- Given point  $p_q$ , find its neighbors within radius  $r$
- PFH considers **each pair** within this ball/circle



Source: PFH PCL, [http://pointclouds.org/documentation/tutorials/pfh\\_estimation.php](http://pointclouds.org/documentation/tutorials/pfh_estimation.php)



- Define a local reference frame that is independent to 6D pose.

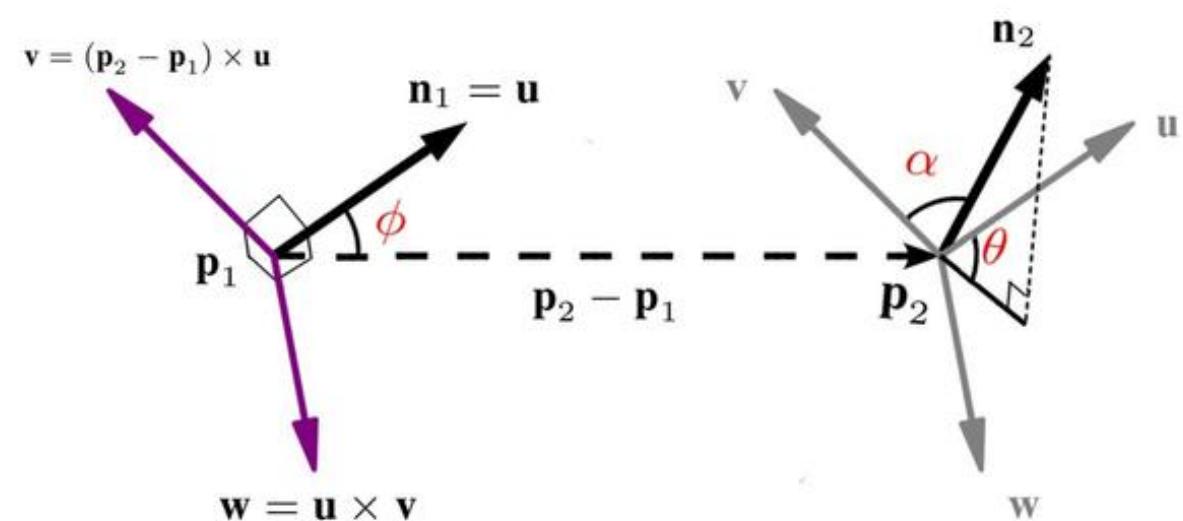
$n_1$ : surface normal at  $p_1$

$n_2$ : surface normal at  $p_2$

$$u = n_1$$

$$v = u \times \frac{p_2 - p_1}{\|p_2 - p_1\|_2}$$

$$w = u \times v$$





- Goal: difference between surface normal  $n_1, n_2 \rightarrow [\alpha, \phi, \theta, d]$

$$d = \|p_2 - p_1\|_2$$

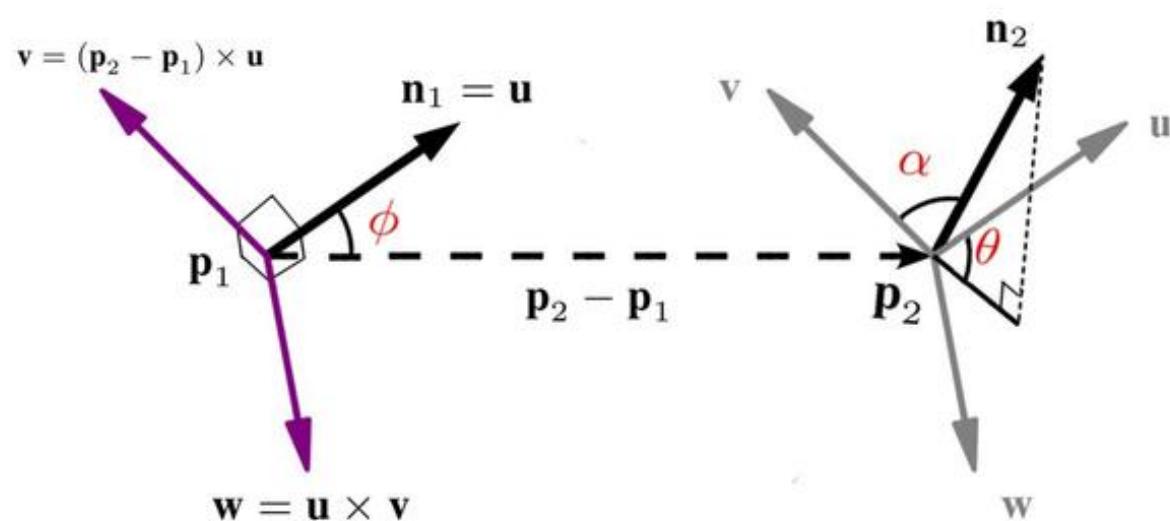
$$\alpha = v \cdot n_2$$

$$\phi = u \cdot \frac{p_2 - p_1}{\|p_2 - p_1\|_2}$$

$$\theta = \arctan(w \cdot n_2, u \cdot n_2)$$

$$n_2 \text{ projection on } w: \frac{n_2 \cdot w}{\|w\|_2} = w \cdot n_2$$

$n_2$  projection on  $u$



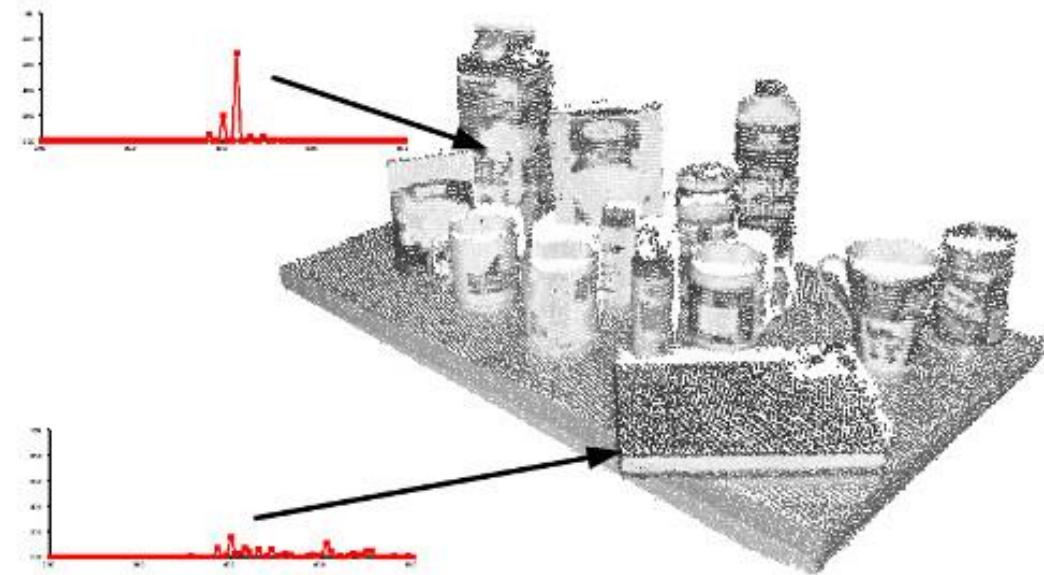
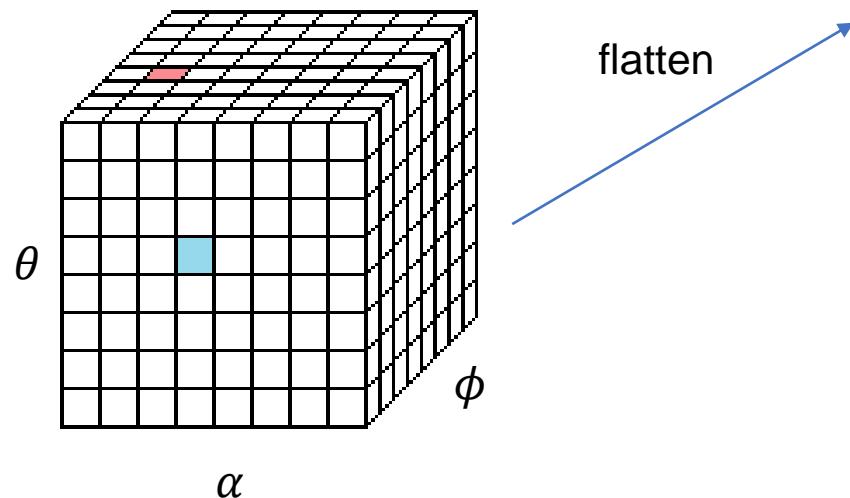


- Each pair of points gives a quadruplet  $[\alpha, \phi, \theta, d]$ 
  - Usually  $d$  is ignored because
    - Usually point densities changes according to viewpoint
    - We don't want the descriptors depends on viewpoint
- If there are  $k$  points in the neighborhood, there are  $k^2$  quadruplets/triplets
- Put the  $k^2$  triplets  $[\alpha, \phi, \theta]$  into histograms
  - Treat each triplet as a 3D data point
  - Each dimension has  $B$  bins
  - Put each triplet into this 3D “voxel grid”
  - The flatten “voxel grid” is a  $B^3$  array - PFH feature vector
  - Normalize, e.g., sum/norm equals to some value.



## PFH Example – Different regions has different histograms

Put the  $k^2$  triplets  $[\alpha, \phi, \theta]$  into histograms



Source: PFH PCL, [http://pointclouds.org/documentation/tutorials/pfh\\_estimation.php](http://pointclouds.org/documentation/tutorials/pfh_estimation.php)



## PFH Example – Semantic Segmentation

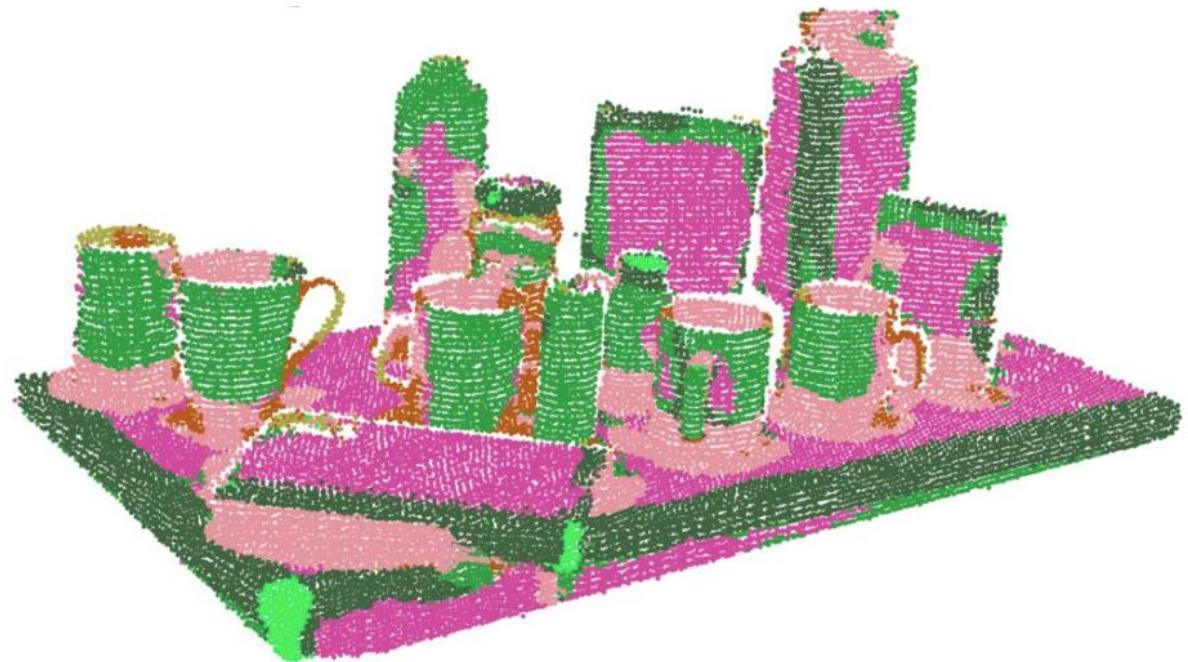
Semantic segmentation



Per-point classification



Per-point PFH + SVM



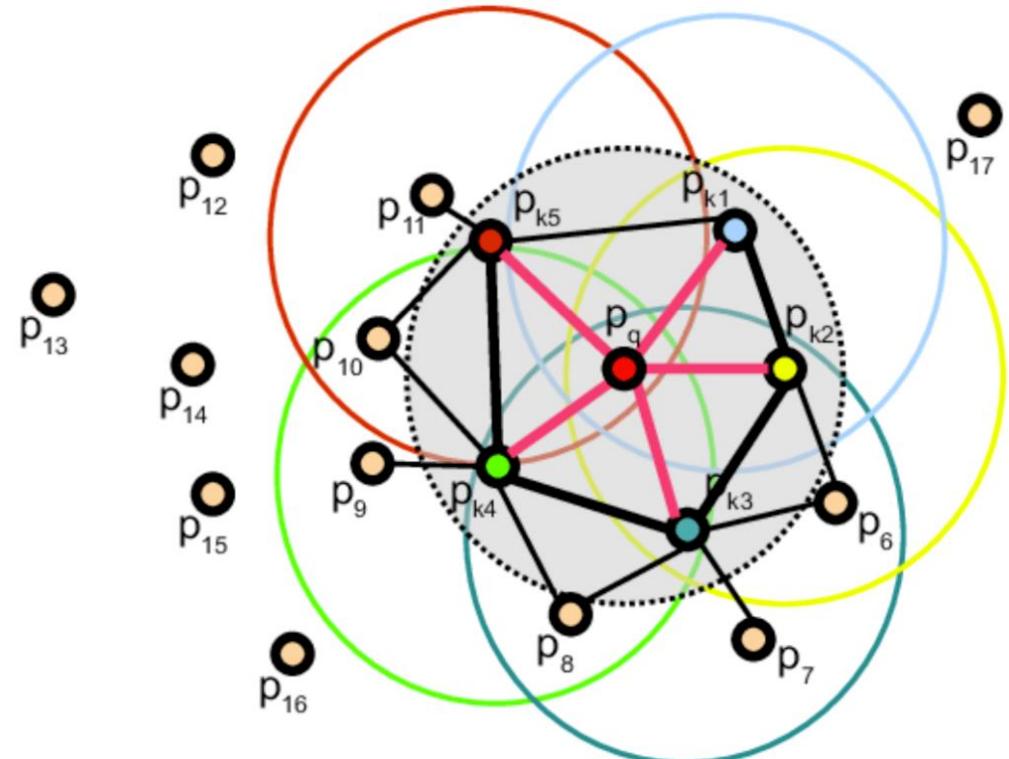


- Idea:
  - 6D-Pose independent
    - Create a local reference frame based on surface normal.
  - Captures surface variations in a neighborhood
    - Compute the surface normal difference for each pair of points
- Complexity:
  - For each keypoint  $p_q$  with  $k$  points in the neighborhood:  $O(k^2)$
  - $n$  points:  $O(nk^2)$
  - Can be improved to be  $O(k)/O(nk)$  - FPFH
- Characteristics
  - Simple and effective
  - Sensitive to surface normal estimation



## FPFH – Fast Point Feature Histogram

- Simplified Point Feature Histogram (SPFH)
  - Compute triplet  $[\alpha, \phi, \theta]$  between query point and its neighbors within  $r$
  - PFH – compute triplet between every pair within  $r$
- Output is 3 histograms (each has  $B$  bins) by binning triplet  $[\alpha, \phi, \theta]$



Source: FPFH PCL, [http://pointclouds.org/documentation/tutorials/fpfh\\_estimation.php](http://pointclouds.org/documentation/tutorials/fpfh_estimation.php)

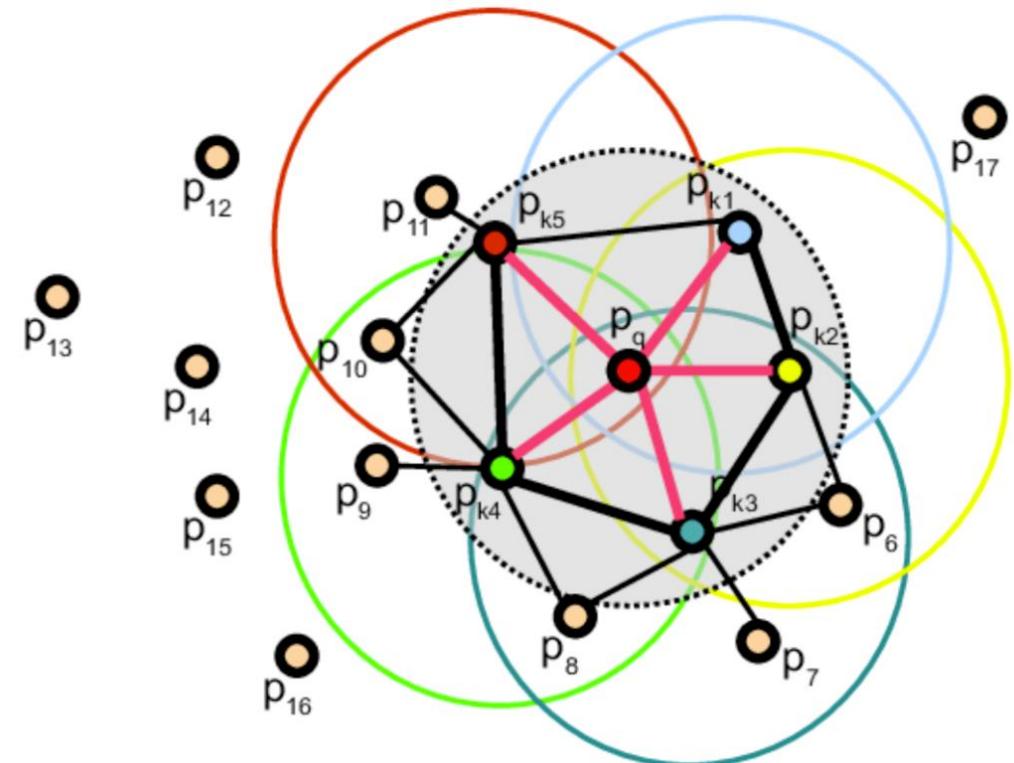


## FPFH – Fast Point Feature Histogram

- FPFH is the weighted sum of neighboring SPFH
  1. Compute SPFH of query point
  2. Compute SPFH of neighbor points
  3. FPFH = weighted sum of (1)(2)

$$FPFH(p_q) = SPFH(p_q) + \frac{1}{k} \sum_{i=1}^k w_k \cdot SPFH(p_k)$$

$$w_k = \frac{1}{\|p_q - p_k\|_2}$$



Source: FPFH PCL, [http://pointclouds.org/documentation/tutorials/fpfh\\_estimation.php](http://pointclouds.org/documentation/tutorials/fpfh_estimation.php)

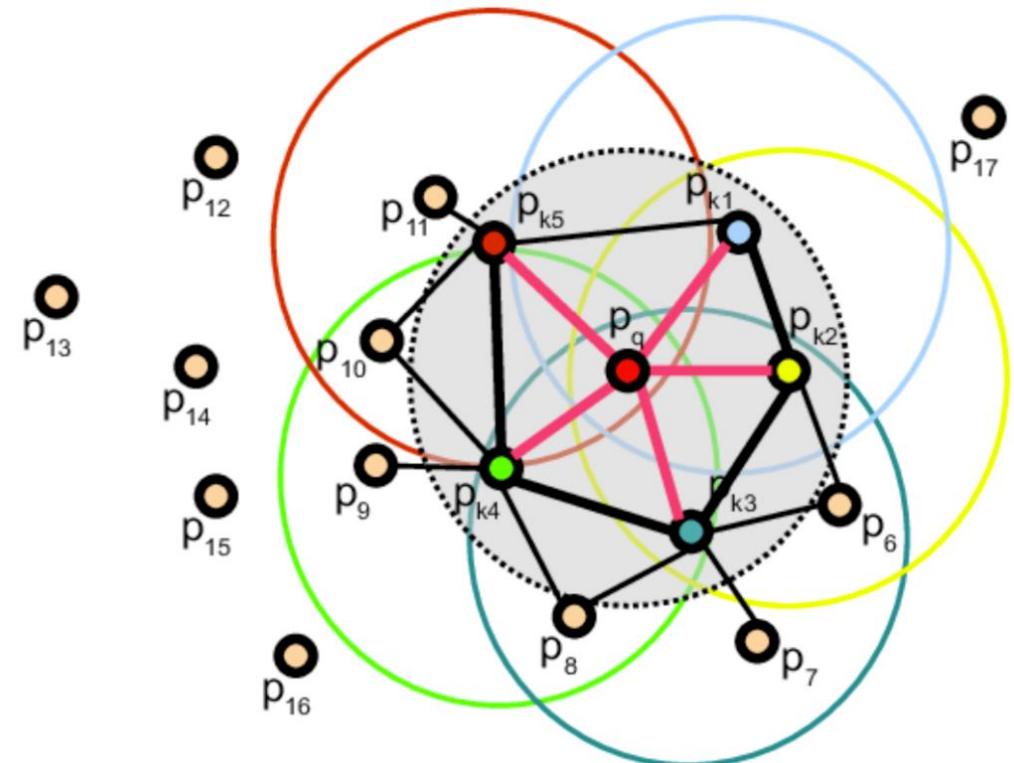


## FPFH – Fast Point Feature Histogram

- FPFH is the weighted sum of neighboring SPFH
- Some edges are counted twice (thick edges)
- 3 Histograms are **concatenated**, not “voxel grid”.

$$FPFH(p_q) = SPFH(p_q) + \frac{1}{k} \sum_{i=1}^k w_k \cdot SPFH(p_k)$$

$$w_k = \frac{1}{\|p_q - p_k\|_2}$$

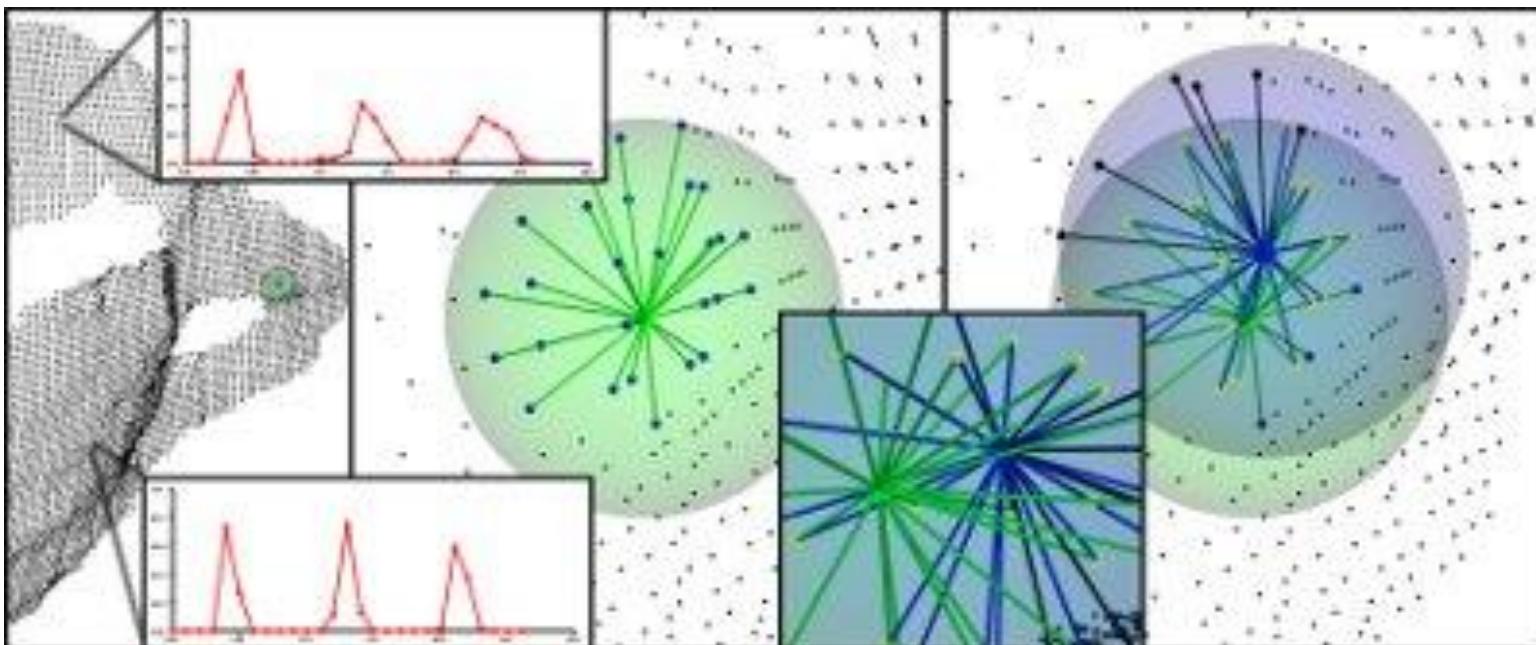


Source: FPFH PCL, [http://pointclouds.org/documentation/tutorials/fpfh\\_estimation.php](http://pointclouds.org/documentation/tutorials/fpfh_estimation.php)



## FPFH Example

- In this example, there are 3 peaks in the FPFH histogram
- Because it is concatenation of 3 histograms. Each histogram has a peak in this example.



Source: FPFH PCL, [http://pointclouds.org/documentation/tutorials/fpfh\\_estimation.php](http://pointclouds.org/documentation/tutorials/fpfh_estimation.php)



### FPFH

- Partial connected neighbors
- Neighborhood of range  $[r, 2r]$
- Some edge are counted twice
- $O(nk)$
- Histogram size  $3B$

### PFH

- Fully connected neighbors
- Neighborhood of range  $r$
- Each edge is counted once
- $O(nk^2)$
- Histogram size  $B^3$



- PFH / FPFH encodes **pair-wise** information with  $\alpha, \phi, \theta$ 
  - 6D-Pose independent
  - Captures surface variations in a neighborhood
  - Neighbor positions are not directly recorded
- Why don't we encode **neighborhood position** information?
  - E.g., 5 points on 6 o'clock direction, 8 points on 9 o'clock direction, etc.
  - But local coordinate has to be **6D-Pose independent**.
  - Solution: build a canonical pose of the local neighborhood
    - Local Reference Frame (LRF)



1. Weighted covariance matrix within radius  $R$

$$\mathbf{M} = \frac{1}{\sum_{i:d_i \leq R} (R - d_i)} \sum_{i:d_i \leq R} (R - d_i)(\mathbf{p}_i - \mathbf{p})(\mathbf{p}_i - \mathbf{p})^T \quad d_i = \|\mathbf{p}_i - \mathbf{p}\|_2$$

2. Compute the three eigenvectors in decreasing eigenvalue order

- Denoted as  $\mathbf{x}^+, \mathbf{y}^+, \mathbf{z}^+$
- Opposite direction denoted as  $\mathbf{x}^-, \mathbf{y}^-, \mathbf{z}^-$

3. Determine  $\mathbf{x}$

$$S_x^+ \doteq \{i : d_i \leq R \wedge (\mathbf{p}_i - \mathbf{p}) \cdot \mathbf{x}^+ \geq 0\} \quad \text{Number of points on the half space of } \mathbf{x}^+$$

$$S_x^- \doteq \{i : d_i \leq R \wedge (\mathbf{p}_i - \mathbf{p}) \cdot \mathbf{x}^- > 0\} \quad \text{Number of points on the half space of } \mathbf{x}^-$$

$$\mathbf{x} = \begin{cases} \mathbf{x}^+, & |S_x^+| \geq |S_x^-| \\ \mathbf{x}^-, & \text{otherwise} \end{cases}$$

4. Determine  $\mathbf{z}$  similarly.

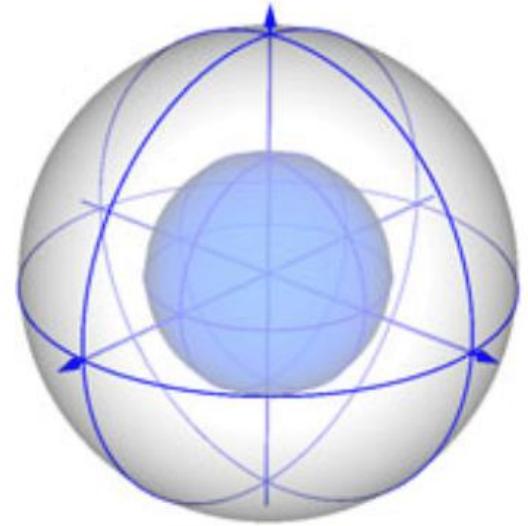
5.  $\mathbf{y} = \mathbf{z} \times \mathbf{x}$

PCA? There is the positive / negative ambiguity –  
each principle vector has two directions



## SHOT - Signatures of Histogram

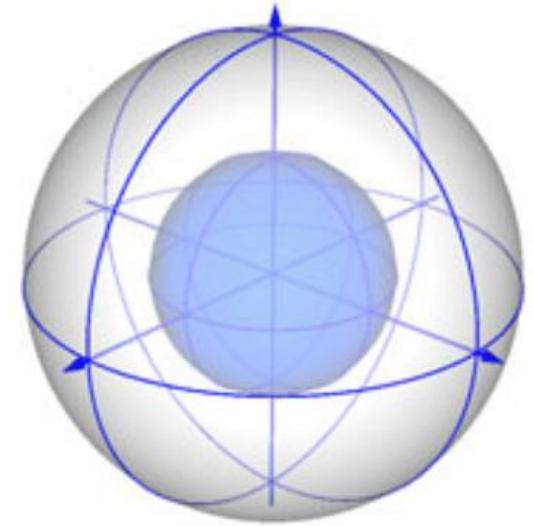
- With the LRF, we can
  1. Divide the space into several small volumes
  2. Compute local histogram of each volume
  3. Concatenate local histograms into a "signature"
    - With LRF, the signature is 6D pose invariant
  4. Normalize the "signature" into sum=1. This is the descriptor





## SHOT - Signatures of Histogram

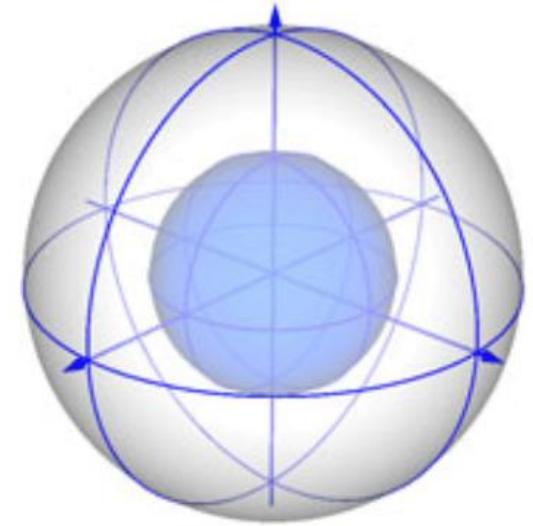
1. Division into **32** volumes
  - 8 azimuth divisions
    - Figure on the right shows only 4 azimuth divisions for clarity
  - 2 elevation divisions
  - 2 radial divisions
2. Build a histogram of  $\cos\theta_i$  **for each volume**
  - Surface normal of a point in that volume,  $n_{v_i}$
  - Surface normal of the keypoint,  $n_u$
  - $\cos\theta_i = n_u \cdot n_{v_i}$
  - Increase the corresponding bin of  $\cos\theta_i$  for that volume
  - For example, `pcl::SHOT352` builds histogram of length 11 for each volume.  $11 \times 32 = 352$
- **Boundary Effect!**
  - Points at the edge of each volume should contribute to neighboring volume as well
  - Small perturbation of LRF changes all the local histograms





## SHOT – Boundary Effect

- For a point  $x_i = (\rho_i, \alpha_i, \beta_i)$  with computed  $\cos \theta_i$ 
  - $\rho_i$  is distance to keypoint. Division resolution  $r_\rho$
  - $\alpha_i$  is azimuth angle in LRF. Division resolution  $r_\alpha$
  - $\beta_i$  is elevation angle in LRF. Division resolution  $r_\beta$
  - $\cos \theta_i$  is the surface normal dot product. Division resolution  $r_\theta$
  - It contributes to  $2^3 = 8$  volumes
    - $\left(\left\lfloor \frac{\rho_i}{r_\rho} \right\rfloor \text{ and } \left\lfloor \frac{\rho_i}{r_\rho} \right\rfloor, \left\lfloor \frac{\alpha_i}{r_\alpha} \right\rfloor \text{ and } \left\lfloor \frac{\alpha_i}{r_\alpha} \right\rfloor, \left\lfloor \frac{\beta_i}{r_\beta} \right\rfloor \text{ and } \left\lfloor \frac{\beta_i}{r_\beta} \right\rfloor\right)$
    - In each volume, it contributes to 2 bins in the histogram
      - $\left\lfloor \frac{\cos \theta_i}{r_\theta} \right\rfloor \text{ and } \left\lfloor \frac{\cos \theta_i}{r_\theta} \right\rfloor$
  - It is quadrilinear interpolation (interpolation that involves 4 dimensions  $\rho, \alpha, \beta, \cos \theta$ )





## SHOT – Boundary Effect

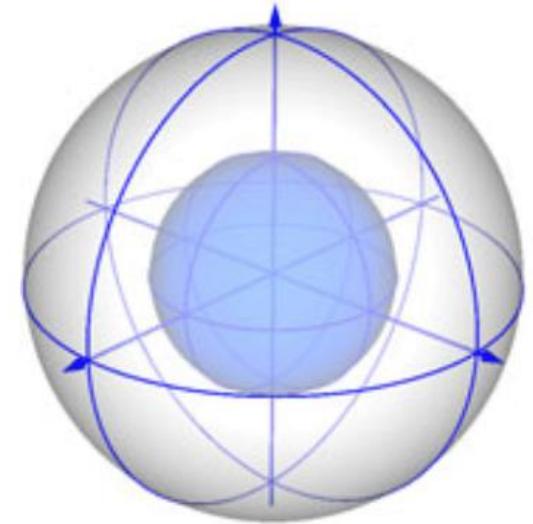
- Let's look at the contribution into

- volume  $\left[ \frac{\rho_i}{r_\rho} \right], \left[ \frac{\alpha_i}{r_\alpha} \right], \left[ \frac{\beta_i}{r_\beta} \right]$
- Histogram bin  $\left[ \frac{\cos \theta_i}{r_\theta} \right]$

- Weighting in each dimension

- $w_\rho = 1 - \frac{\left[ \frac{\rho_i}{r_\rho} \right] r_\rho - \rho_i}{r_\rho}, w_\alpha = 1 - \frac{\alpha_i - \left[ \frac{\alpha_i}{r_\alpha} \right] r_\alpha}{r_\alpha}, w_\beta = 1 - \frac{\beta_i - \left[ \frac{\beta_i}{r_\beta} \right] r_\beta}{r_\beta}, w_\theta = 1 - \frac{\cos \theta_i - \left[ \frac{\cos \theta_i}{r_\theta} \right] r_\theta}{r_\theta}$

- The contribution to that bin at that volume is  $w = w_\rho w_\alpha w_\beta w_\theta$



## Registration Recall on 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1937	0.1779	0.3063	0.5751	0.4605	<b>0.8972</b>	0.5949	0.7352	0.7866
Home 1	0.3974	0.3718	0.5833	0.7372	0.6154	0.5577	0.7179	0.7564	<b>0.7628</b>
Home 2	0.3654	0.3365	0.4663	<b>0.7067</b>	0.5625	0.5913	0.6058	0.6250	0.6154
Hotel 1	0.1814	0.2080	0.2611	0.5708	0.4469	0.5796	0.6549	0.6593	<b>0.6814</b>
Hotel 2	0.2019	0.2212	0.3269	0.4423	0.3846	0.5769	0.4231	0.6058	<b>0.7115</b>
Hotel 3	0.3148	0.3889	0.5000	0.6296	0.5926	0.6111	0.6111	0.8889	<b>0.9444</b>
Study	0.0548	0.0719	0.1541	0.5616	0.4075	0.5342	<b>0.7123</b>	0.5753	0.6199
MIT Lab	0.1039	0.1299	0.2727	0.5455	0.3506	<b>0.6364</b>	0.5844	0.5974	0.6234
Average	0.2267	0.2382	0.3589	0.5961	0.4776	0.6231	0.6130	0.6804	<b>0.7182</b>

## Registration Recall on **Rotated** 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1779	0.1779	0.2905	0.0040	0.4466	0.0020	0.0178	0.7352	<b>0.7885</b>
Home 1	0.4487	0.3526	0.5897	0.0128	0.6667	0.0000	0.0321	0.7692	<b>0.7821</b>
Home 2	0.3413	0.3365	0.4712	0.0337	0.5288	0.0144	0.0337	0.6202	<b>0.6442</b>
Hotel 1	0.1814	0.2168	0.3009	0.0044	0.4425	0.0044	0.0133	0.6637	<b>0.6770</b>
Hotel 2	0.1731	0.2404	0.2981	0.0000	0.4423	0.0000	0.0096	0.6058	<b>0.6923</b>
Hotel 3	0.3148	0.3333	0.5185	0.0096	0.6296	0.0000	0.0370	0.9259	<b>0.9630</b>
Study	0.0582	0.0822	0.1575	0.0000	0.4178	0.0000	0.0171	0.5616	<b>0.6267</b>
MIT Lab	0.1169	0.1299	0.2857	0.0260	0.4156	0.0000	0.0260	0.6104	<b>0.6753</b>
Average	0.2265	0.2337	0.3640	0.0113	0.4987	0.0026	0.0233	0.6865	<b>0.7311</b>



### FPFH

- Partial connected neighbors
- Neighborhood of range  $[r, 2r]$
- Pairwise Reference Frame
- $O(nk)$
- Histogram size  $3B$

### PFH

- Fully connected neighbors
- Neighborhood of range  $r$
- Pairwise Reference Frame
- $O(nk^2)$
- Histogram size  $B^3$

### SHOT

- Only connects a keypoint with its neighbors
- Neighborhood of range  $r$
- Local Reference Frame
- $O(nk)$
- Descriptor size  $32 \times \text{histogram\_size}$

Similar to PPFNet / PPF-FoldeNet

Similar to LRF in PerfectMatch



## Why do we need Deep Learning Descriptor

- All handcrafted descriptors are based on geometry, e.g.,
  - Surface normal variations around the keypoint.
  - Point distributions around the keypoint
- They are not reliable in case of
  - Noise
  - Occlusion / incomplete shape
  - Sparsity
- Deep learning
  - Includes semantic information
  - Smarter way to encode geometry
  - Robust to noise



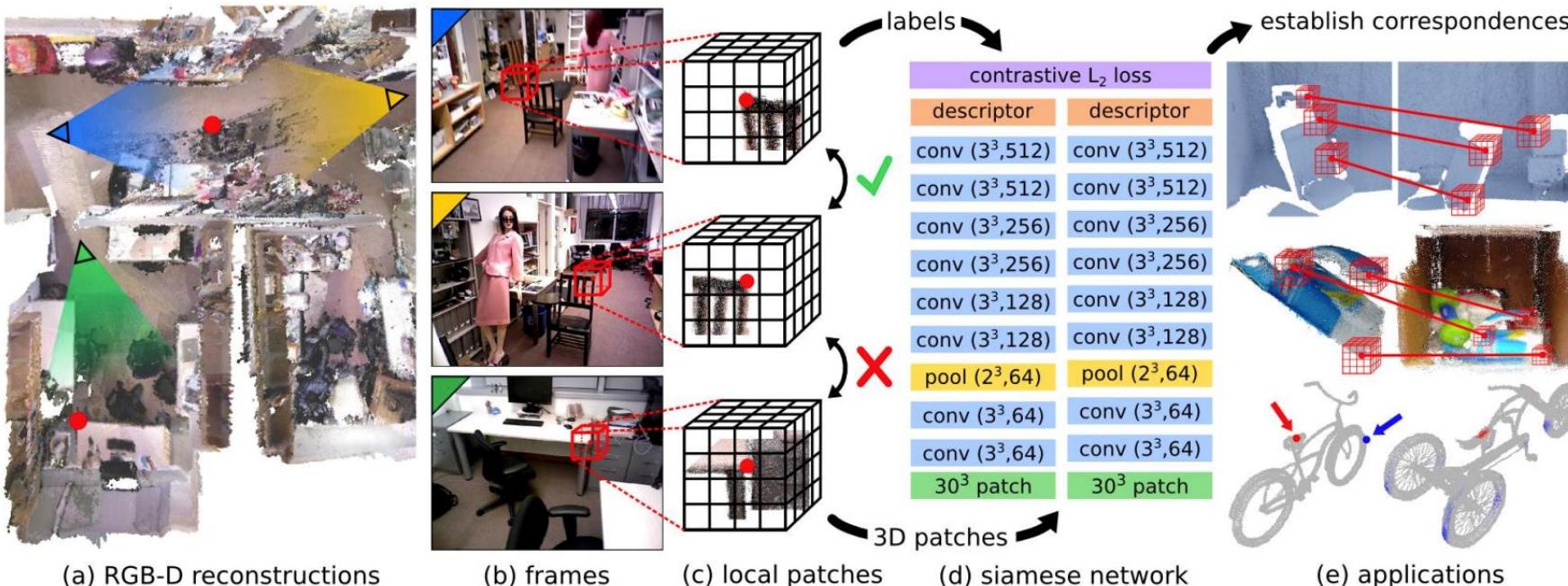
## Point Cloud Feature Descriptor – Deep Learning

- Lots of approaches
- Usually better performance
- Methods:
  - **3DMatch**: Learning Local Geometric Descriptors from RGB-D Reconstructions
  - **The Perfect Match**: 3D Point Cloud Matching with Smoothed Densities
  - **PPFNet**: Global Context Aware Local Features for Robust 3D Point Matching
  - **PPF-FoldNet**: Unsupervised Learning of Rotation Invariant 3D Local Descriptors
  - CGF: Learning Compact Geometric Features
  - 3DFeat-Net: Weakly Supervised Local 3D Features for Point Cloud Registration
  - USIP: Unsupervised Stable Interest Point Detection from 3D Point Clouds
  - ... ...



- Inference

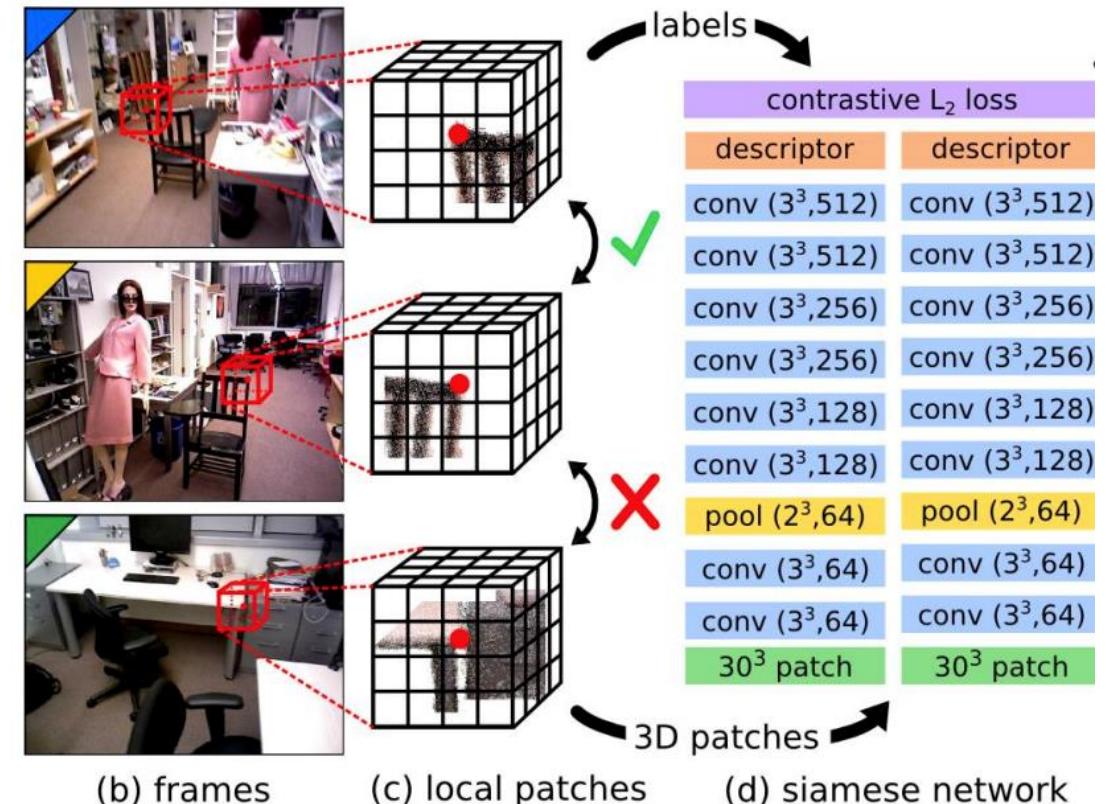
- Input: 3D patch, voxel grid of  $30 \times 30 \times 30$ .
  - Truncated Distance Function (TDF) / Binary Voxel Grid / Probabilistic Voxel Grid, etc.
- Output: Descriptor for that patch, vector of 512





## 3DMatch – Training

- Training:
  - *Input*: two patches
  - *Output*: two descriptors
  - If the two patches are from the same location:
    - Make them similar
  - Else:
    - Make them different.
- Two problems to be solved:
  - How to build dataset
  - How to define “similar” / “different”

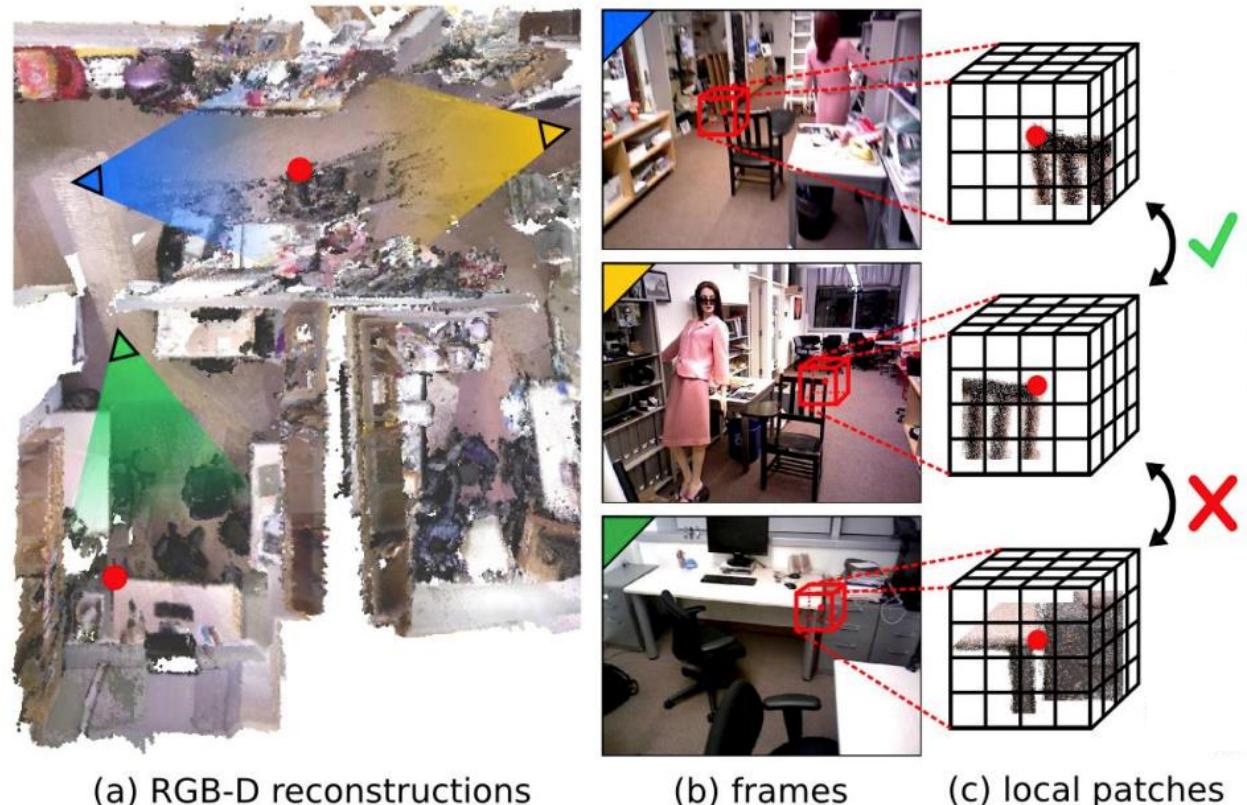


Source: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions, Andy Zeng, et.al.



## 3DMatch – Dataset / Training Correspondences

- Utilize RGB-D reconstructions
  - Reconstructed from multiple RGB-D frames.
  - We know the position of each frame.
- Positive pairs of patches:
  - Same physical location (e.g. <0.05m)
  - Different frames that are captures at least 1m apart.
- Negative pairs:
  - At least 0.1m apart.



Source: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions, Andy Zeng, et.al.



## 3DMatch – Contrastive Loss

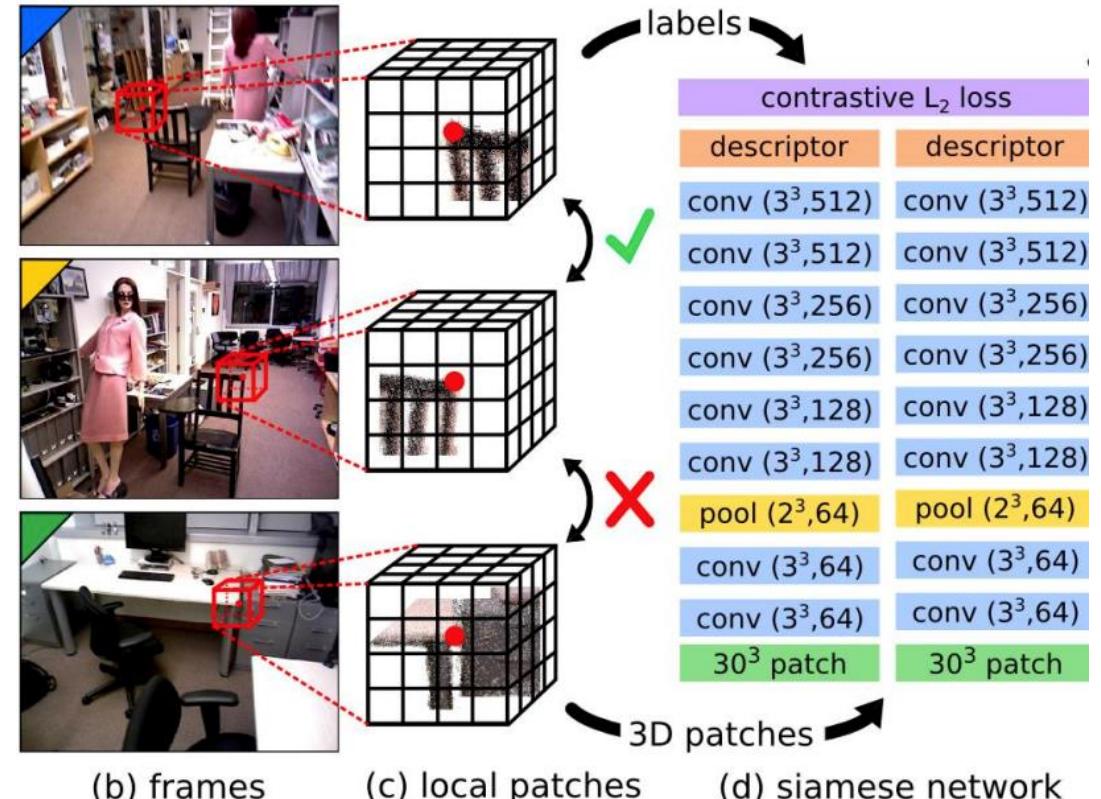
- There is ground truth label  $y_{ij}$

- Positive pairs:  $y_{ij} = 1$
- Negative pairs:  $y_{ij} = 0$

- Contrastive Loss

$$L = \frac{1}{N} \sum_{n=1}^N y_{ij} d_{ij}^2 + (1 - y_{ij}) \max(\tau - d_{ij}, 0)^2$$
$$d_{ij} = \|f(x_i) - f(x_j)\|_2$$

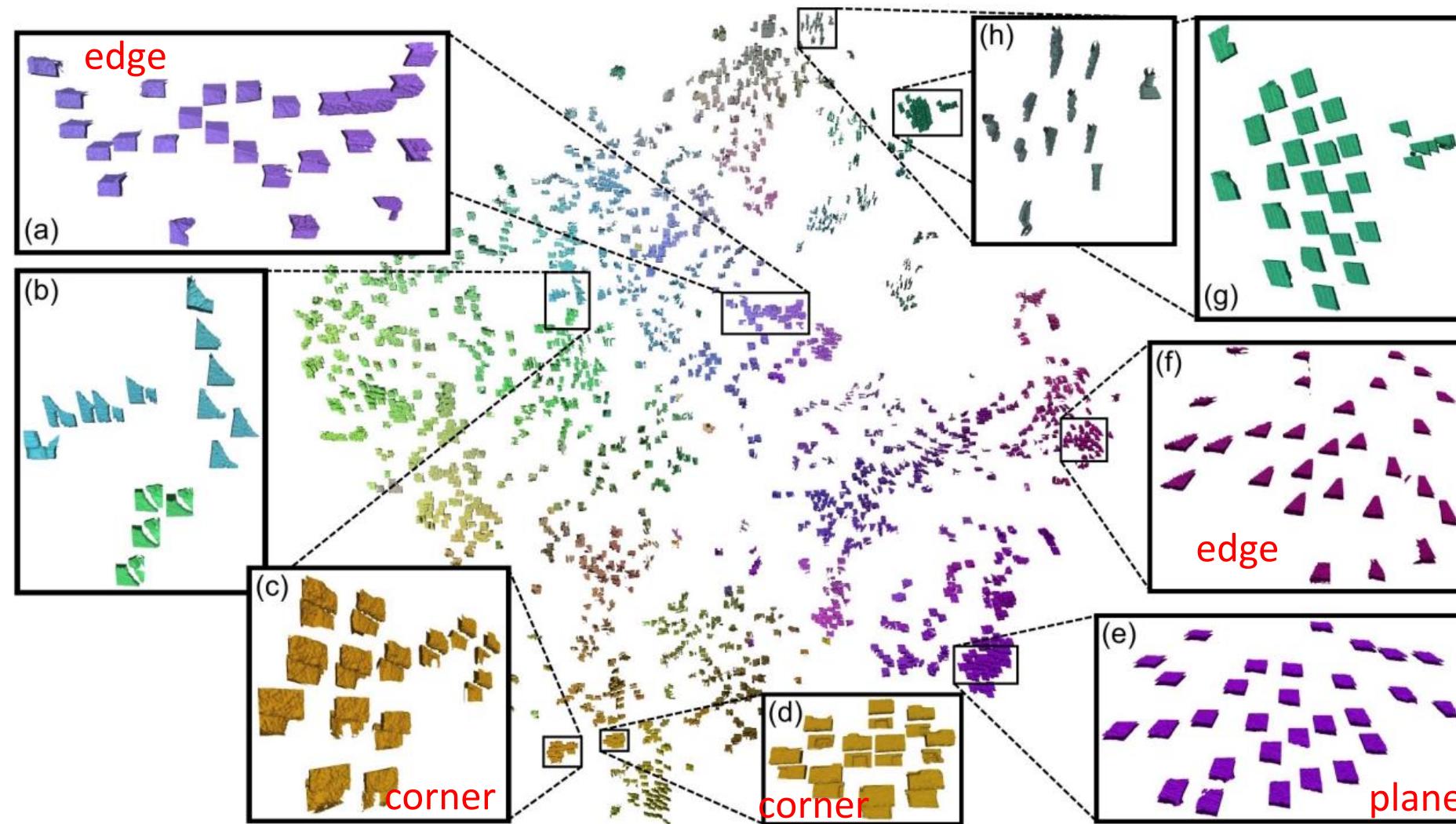
- Positive: Pull them together
- Negative: Push them to  $\tau$  away.
  - $\tau$  is the margin. If the negative pair is already  $\tau$  away, ignore it.



Source: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions, Andy Zeng, et.al.



## 3DMatch – t-SNE embedding of descriptors

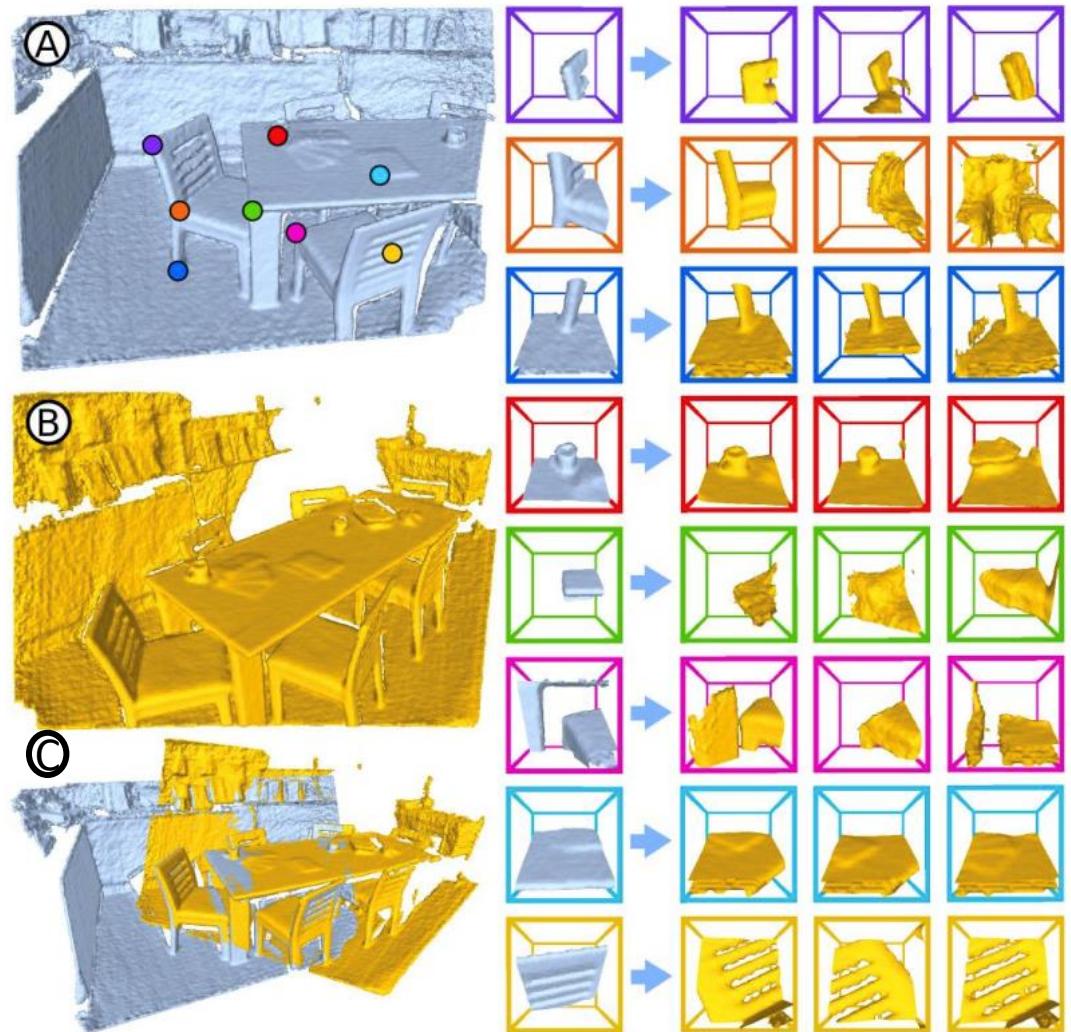


Source: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions, Andy Zeng, et.al.



## 3DMatch – Results

- A, B: RGBD scans from different view point
- C: Registered with 3DMatch + RANSAC (covered in Lecture 9)
- The 3 columns on the right: 3NN search of A's patches in B, based on the 3DMatch descriptors.



Source: 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions, Andy Zeng, et.al.

## Registration Recall on 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1937	0.1779	0.3063	0.5751	0.4605	<b>0.8972</b>	0.5949	0.7352	0.7866
Home 1	0.3974	0.3718	0.5833	0.7372	0.6154	0.5577	0.7179	0.7564	<b>0.7628</b>
Home 2	0.3654	0.3365	0.4663	<b>0.7067</b>	0.5625	0.5913	0.6058	0.6250	0.6154
Hotel 1	0.1814	0.2080	0.2611	0.5708	0.4469	0.5796	0.6549	0.6593	<b>0.6814</b>
Hotel 2	0.2019	0.2212	0.3269	0.4423	0.3846	0.5769	0.4231	0.6058	<b>0.7115</b>
Hotel 3	0.3148	0.3889	0.5000	0.6296	0.5926	0.6111	0.6111	0.8889	<b>0.9444</b>
Study	0.0548	0.0719	0.1541	0.5616	0.4075	0.5342	<b>0.7123</b>	0.5753	0.6199
MIT Lab	0.1039	0.1299	0.2727	0.5455	0.3506	<b>0.6364</b>	0.5844	0.5974	0.6234
Average	0.2267	0.2382	0.3589	0.5961	0.4776	0.6231	0.6130	0.6804	<b>0.7182</b>

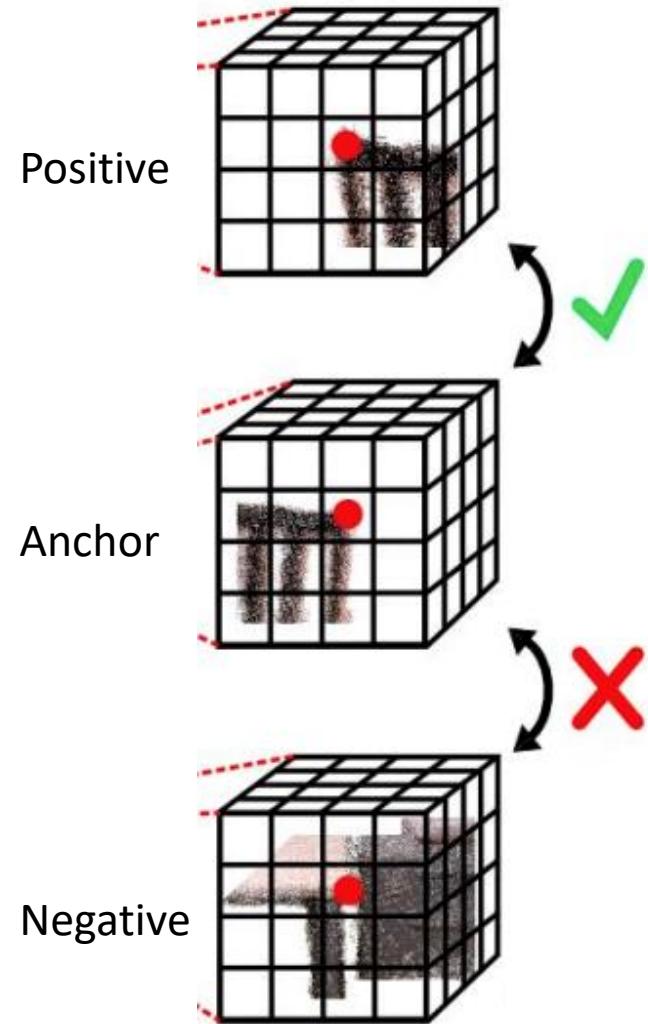
## Registration Recall on **Rotated** 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1779	0.1779	0.2905	0.0040	0.4466	0.0020	0.0178	0.7352	<b>0.7885</b>
Home 1	0.4487	0.3526	0.5897	0.0128	0.6667	0.0000	0.0321	0.7692	<b>0.7821</b>
Home 2	0.3413	0.3365	0.4712	0.0337	0.5288	0.0144	0.0337	0.6202	<b>0.6442</b>
Hotel 1	0.1814	0.2168	0.3009	0.0044	0.4425	0.0044	0.0133	0.6637	<b>0.6770</b>
Hotel 2	0.1731	0.2404	0.2981	0.0000	0.4423	0.0000	0.0096	0.6058	<b>0.6923</b>
Hotel 3	0.3148	0.3333	0.5185	0.0096	0.6296	0.0000	0.0370	0.9259	<b>0.9630</b>
Study	0.0582	0.0822	0.1575	0.0000	0.4178	0.0000	0.0171	0.5616	<b>0.6267</b>
MIT Lab	0.1169	0.1299	0.2857	0.0260	0.4156	0.0000	0.0260	0.6104	<b>0.6753</b>
Average	0.2265	0.2337	0.3640	0.0113	0.4987	0.0026	0.0233	0.6865	<b>0.7311</b>



## How to Improve 3DMatch?

- Contrastive Loss is too "greedy".
  - $L = \frac{1}{N} \sum_{n=1}^N y_{ij} d_{ij}^2 + (1 - y_{ij}) \max(\tau - d_{ij}, 0)^2$
- Another representation of contrastive loss:
  - **Anchor** sample  $x_i^a$
  - **Positive** sample  $x_i^p$  - same location
  - **Negative** sample  $x_i^n$  - different location
  - Assume equal number of positive and negative pairs
  - $L = \frac{1}{N} \left( \sum_{i=1}^N \|f(x_i^a) - f(x_i^p)\|_2^2 + \sum_{i=1}^N \max(\tau - \|f(x_i^a) - f(x_i^n)\|_2^2, 0) \right)$
- It considers two samples only, either **A+P** or **A+N**

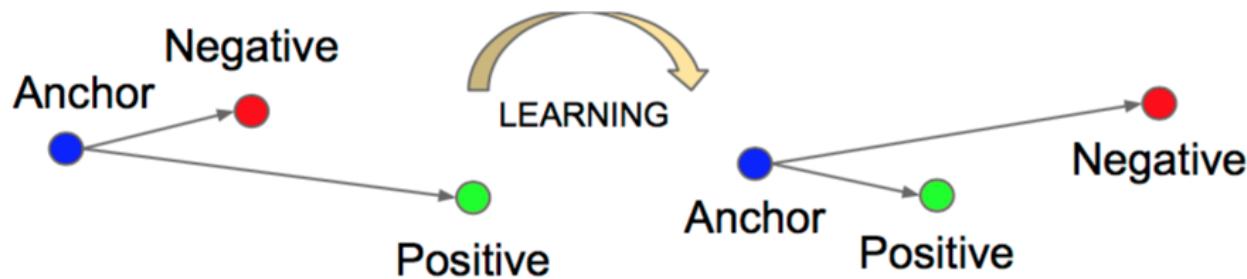




- "Far" and "Close" is relative
  - It is sufficient to distinguish positive and negative when  $d(a, n) \gg d(a, p)$
  - Not necessary to ensure  $d(a, p) \sim 0, d(a, n) > \tau$
- Triplet Loss:
  - Pull A, P together, push A, N away

$$L = \sum_{i=1}^N L_i = \sum_{i=1}^N \max(d_i(a, p) - d_i(a, n) + \gamma, 0)$$

$\gamma$  is the margin





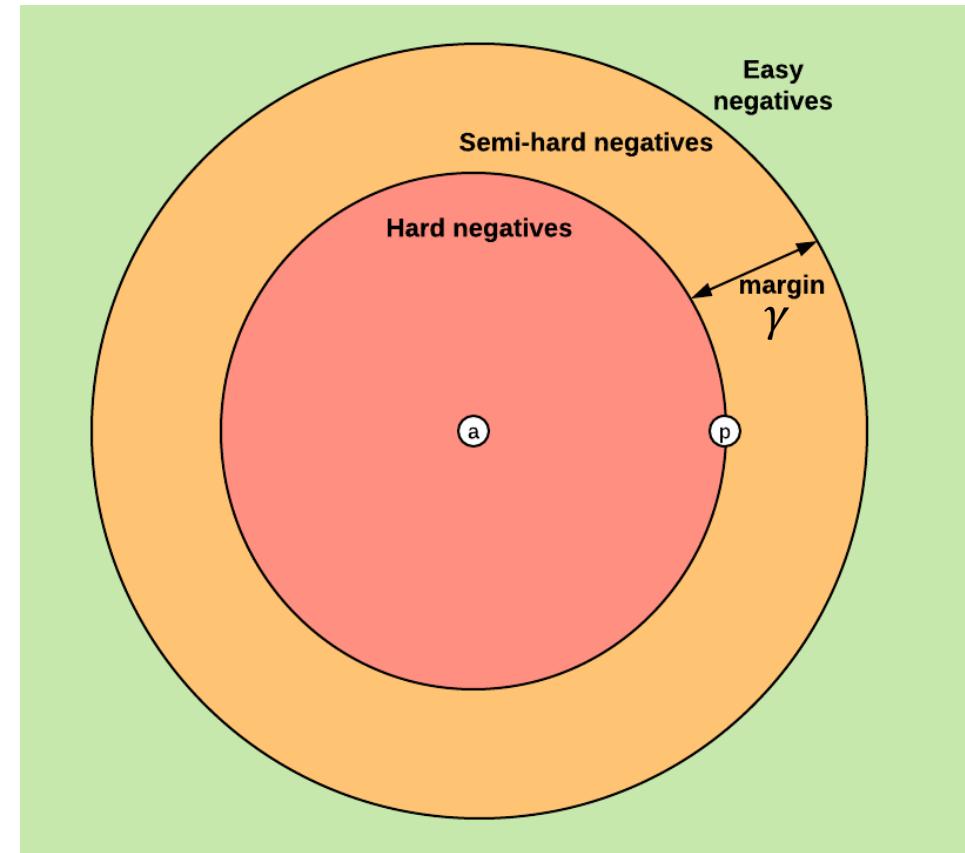
- A triplet loss involves 3 samples: anchor, positive, negative

$$L_i = \max(d_i(a, p) - d_i(a, n) + \gamma, 0)$$

- Problem 1 –  $L_i \approx 0$ , not supervising the network.
  - For each anchor, randomly sample a negative sample.
  - In most cases, that randomly sampled negative is very different to anchor - easy to distinguish
  - $L_i$  is close to 0 in most cases!
- Problem 2 –  $L_i > \gamma$ , network not converging.
  - For each anchor, always find the most similar negative sample, i.e.,  $d_i(a, n) \approx 0$



- Easy triplet
  - Triplet loss is close to 0
  - $d(a, p) - d(a, n) + \gamma < 0$
- Hard triplet
  - The negative is closer to the anchor than the positive
  - $d(a, n) < d(a, p)$
- Semi-hard triplet
  - The positive is closer to the anchor than the negative, but not too much
  - $d(a, p) < d(a, n) < d(a, p) + \gamma$





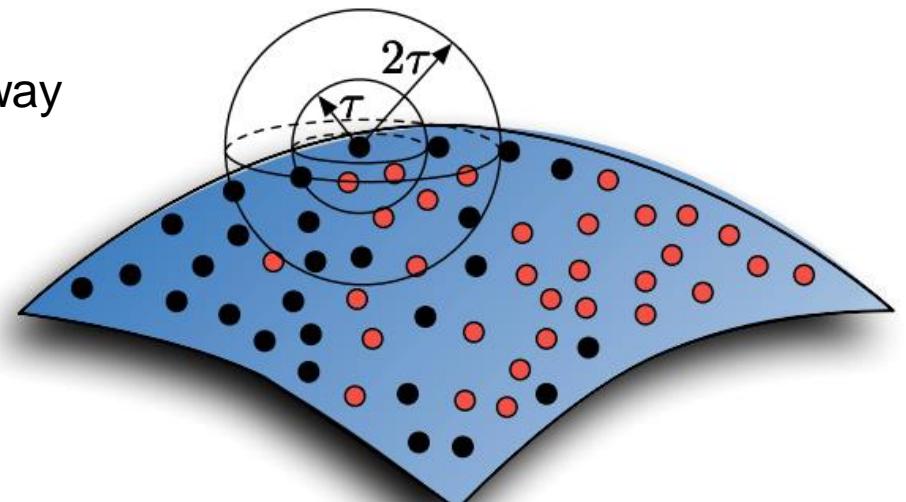
- General guideline
  - Usually triplets should contain semi-hard and/or hard ones
  - Experiment with validation/test set to determine the best configuration and  $\gamma$
  - *In Defense of the Triplet Loss for Person Re-Identification*, Alexander Hermans et.al.
- Offline negative mining – Not efficient
  - Pre-build a list of  $[a, p, n]$
  - After each training epoch
    - Go through the dataset
    - Pick hard positive & negative samples for each anchor



- Batch size B
- 1. There are B pairs of  $a, p$ 
  - $p$  is determined by
    - Getting same location from different view point
    - Randomly getting a patch from the same geometric location (within some small distance from  $a$ )
  - Optional: select  $p$  from frames that is some distance away
  - Not necessary to mine the hardest positives
- 2. Randomly sample B negatives  $n$ 
  - Far away from  $a$
- 3. Feed the 3B samples into network, get 3B descriptors
- 4. For each anchor  $a$ 
  - Find hard negatives by getting  $n^* = \operatorname{argmin} d(a, n)$
- 5. Compute Triplet loss  $L_i = \max(d_i(a, p) - d_i(a, n^*) + \gamma, 0)$



- Batch size B
  1. There are B pairs of  $a, p$ 
    - $p$  is determined by randomly getting a patch from the same geometric location (within some small distance  $\tau$  from  $a$ )
    - Optional: select  $p$  from frames that is some distance away
  2. Select B negatives  $n$ 
    - Distance from  $a$  is  $[\tau, 2\tau]$
  3. Compute Triplet loss
$$L_i = \max(d_i(a, p) - d_i(a, n) + \gamma, 0)$$



## Registration Recall on 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1937	0.1779	0.3063	0.5751	0.4605	<b>0.8972</b>	0.5949	0.7352	0.7866
Home 1	0.3974	0.3718	0.5833	0.7372	0.6154	0.5577	0.7179	0.7564	<b>0.7628</b>
Home 2	0.3654	0.3365	0.4663	<b>0.7067</b>	0.5625	0.5913	0.6058	0.6250	0.6154
Hotel 1	0.1814	0.2080	0.2611	0.5708	0.4469	0.5796	0.6549	0.6593	<b>0.6814</b>
Hotel 2	0.2019	0.2212	0.3269	0.4423	0.3846	0.5769	0.4231	0.6058	<b>0.7115</b>
Hotel 3	0.3148	0.3889	0.5000	0.6296	0.5926	0.6111	0.6111	0.8889	<b>0.9444</b>
Study	0.0548	0.0719	0.1541	0.5616	0.4075	0.5342	<b>0.7123</b>	0.5753	0.6199
MIT Lab	0.1039	0.1299	0.2727	0.5455	0.3506	<b>0.6364</b>	0.5844	0.5974	0.6234
Average	0.2267	0.2382	0.3589	0.5961	0.4776	0.6231	0.6130	0.6804	<b>0.7182</b>

## Registration Recall on **Rotated** 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1779	0.1779	0.2905	0.0040	0.4466	0.0020	0.0178	0.7352	<b>0.7885</b>
Home 1	0.4487	0.3526	0.5897	0.0128	0.6667	0.0000	0.0321	0.7692	<b>0.7821</b>
Home 2	0.3413	0.3365	0.4712	0.0337	0.5288	0.0144	0.0337	0.6202	<b>0.6442</b>
Hotel 1	0.1814	0.2168	0.3009	0.0044	0.4425	0.0044	0.0133	0.6637	<b>0.6770</b>
Hotel 2	0.1731	0.2404	0.2981	0.0000	0.4423	0.0000	0.0096	0.6058	<b>0.6923</b>
Hotel 3	0.3148	0.3333	0.5185	0.0096	0.6296	0.0000	0.0370	0.9259	<b>0.9630</b>
Study	0.0582	0.0822	0.1575	0.0000	0.4178	0.0000	0.0171	0.5616	<b>0.6267</b>
MIT Lab	0.1169	0.1299	0.2857	0.0260	0.4156	0.0000	0.0260	0.6104	<b>0.6753</b>
Average	0.2265	0.2337	0.3640	0.0113	0.4987	0.0026	0.0233	0.6865	<b>0.7311</b>



## How to Make Descriptor Robust to Rotation

- Point cloud / Voxel grid is different after rotation.
- Solution: rotate the patch to canonical representation, i.e., LRF
  - SHOT has a LRF
  - **The Perfect Match:** 3D Point Cloud Matching with Smoothed Densities



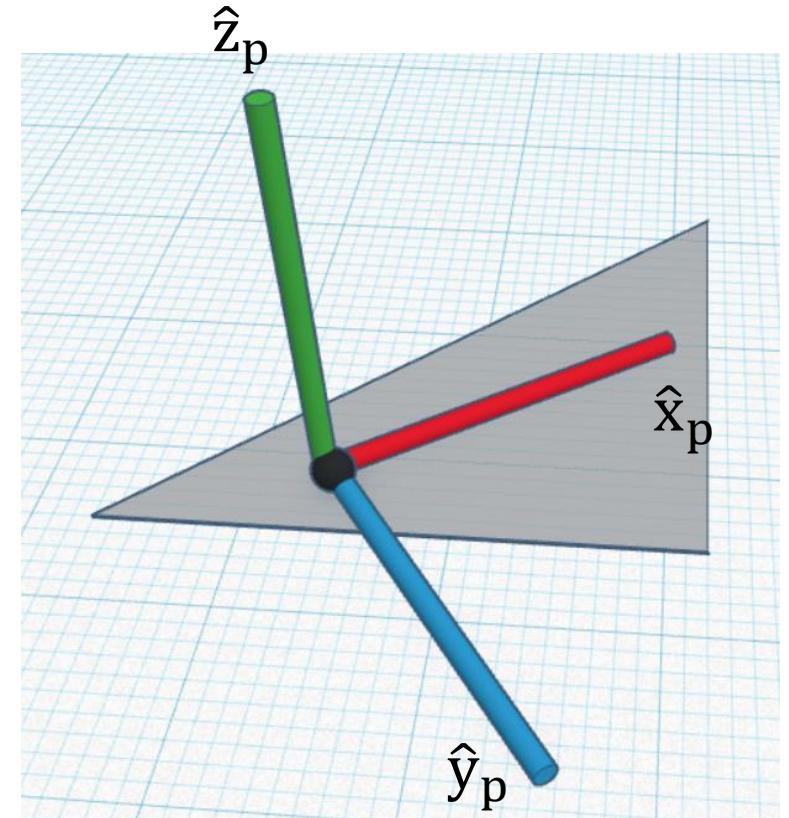
## Local Reference Frame (LRF)

1. Compute covariance matrix around interest point  $p$ 
  - The local support (neighborhood) is defined as  $S = \{p_i : \|p_i - p\|_2 < r\}$

$$\tilde{\Sigma}_S = \frac{1}{|S|} \sum_{p_i \in S} (\mathbf{p}_i - \mathbf{p})(\mathbf{p}_i - \mathbf{p})^T$$

2. Determine  $\hat{z}_p$ 
  - $\hat{n}_p$  is the surface normal
  - Determine direction of  $\hat{z}_p$  (e.g., pointing up or down)
  - Select the direction that sum of  $p_i$  projection is positive.

$$\hat{z}_p = \begin{cases} \hat{n}_p, & \text{if } \sum_{p_i \in S} \langle \hat{n}_p, \overrightarrow{p_i p} \rangle \geq 0 \\ -\hat{n}_p, & \text{otherwise} \end{cases}$$





## Local Reference Frame (LRF)

### 3. Determine $\hat{x}_p$

- Project  $p_i$  onto the plane, get the weighted average as  $\hat{x}_p$

$$\hat{\mathbf{x}}_p = \frac{1}{\left\| \sum_{\mathbf{p}_i \in \mathcal{S}} \alpha_i \beta_i \mathbf{v}_i \right\|_2} \sum_{\mathbf{p}_i \in \mathcal{S}} \alpha_i \beta_i \mathbf{v}_i$$

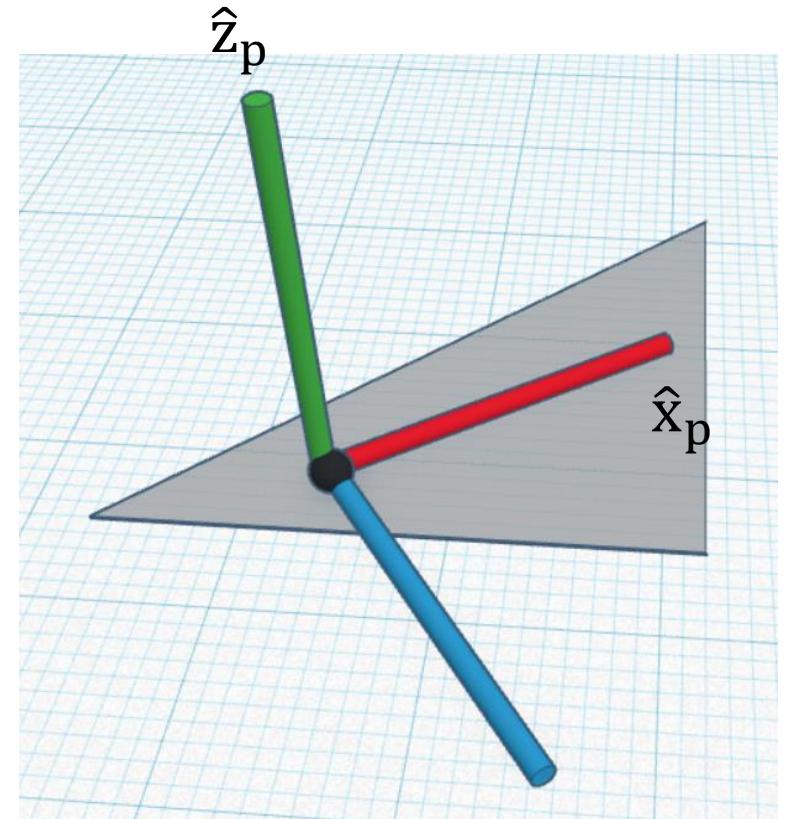
$$\underline{\mathbf{v}_i} = \overrightarrow{\mathbf{p}\mathbf{p}_i} - \langle \overrightarrow{\mathbf{p}\mathbf{p}_i}, \hat{\mathbf{z}}_p \rangle \hat{\mathbf{z}}_p$$

Projection of  $p_i$  on to plane

Projection of  $p_i$  on to  $\hat{z}_p$

$$\alpha_i = (r_{LRF} - \|\mathbf{p} - \mathbf{p}_i\|_2)^2 \quad \text{Closer point} \rightarrow \text{higher weights}$$

$$\beta_i = \langle \overrightarrow{\mathbf{p}\mathbf{p}_i}, \hat{\mathbf{z}}_p \rangle^2 \quad \text{Points further away from plane} \rightarrow \text{higher weights}$$



$$\hat{\mathbf{y}}_p = \hat{\mathbf{x}}_p \times \hat{\mathbf{z}}_p$$



## Local Reference Frame – The Perfect Match

Registration recall  
(higher means better descriptor matching)

	<i>3DMatch</i> data set					
	Original		Rotated			
	Average	STD	Average	STD		
FPFH [28]	54.3	11.8	54.8	12.1		
SHOT [38]	73.3	7.7	73.3	7.6		
3DMatch [49] <sup>2</sup>	57.3	7.8	3.6	1.7		
CGF [17]	58.2	14.2	58.5	14.0		
PPFNet [5]	62.3	11.5	0.3	0.5		
PPF-FoldNet [4]	71.8	9.9	73.1	11.1		
Ours (16 dim)	92.8	3.4	93.0	3.2		
Ours (32 dim)	<b>94.7</b>	2.7	<b>94.9</b>	2.5		

The Perfect Match

Smooth Density Value: another way to fill the voxel grid. (3DMatch utilizes Truncated Distance Function, TDF)



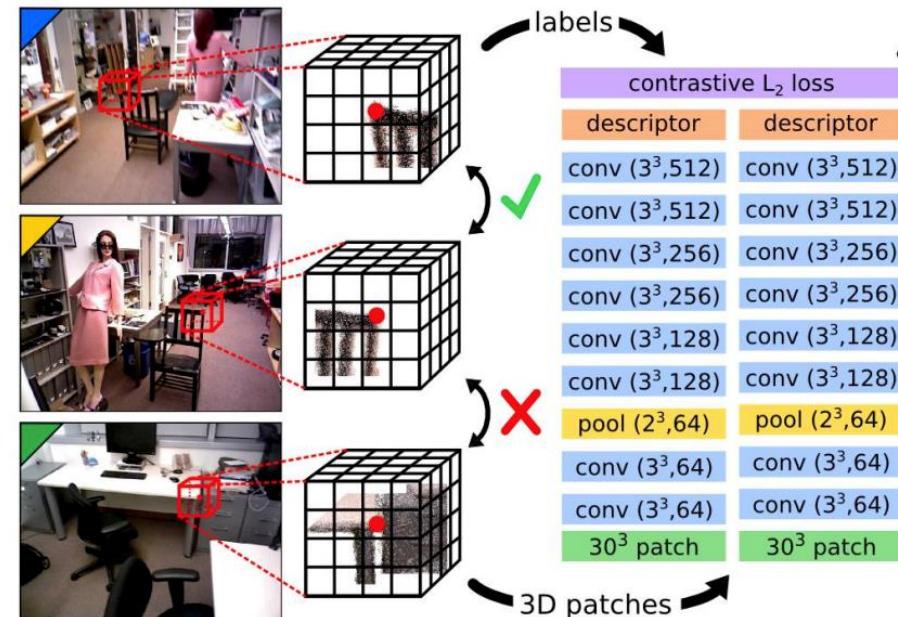
	<i>3DMatch</i> data set			
	Original		Rotated	
	$\tau_2 = 0.05$	$\tau_2 = 0.2$	$\tau_2 = 0.05$	$\tau_2 = 0.2$
All together	94.7	72.7	<b>94.9</b>	<b>72.8</b>
W/o SDV	92.5	63.5	92.5	63.6
W/o LRF	<b>96.3</b>	<b>81.6</b>	11.6	2.7
W/o SDV & LRF	95.6	78.6	9.7	2.1

LRF is the key to make descriptors rotational robust

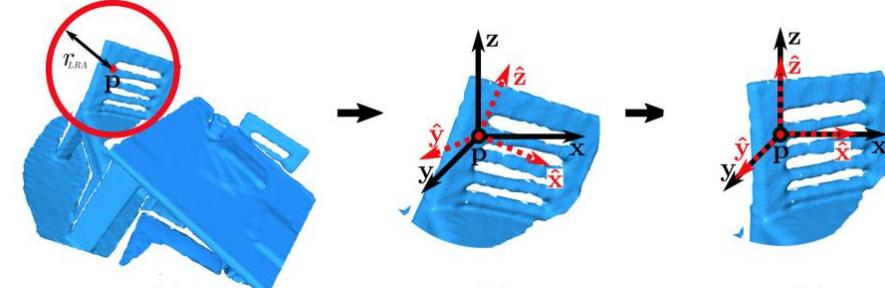
Source: The Perfect Match: 3D Point Cloud Matching with Smoothed Densities, Zan Gojcic, et.al.



- Extract patches around keypoints
  - Truncated Distance Function (TDF)
  - Smooth Density Value (SDV)
  - ...
- Processing
  - Local Reference Frame (LRF)
  - 3D convolutions
- Loss function
  - Contrastive loss
  - Triplet loss
    - Semi-hard / hard triplet mining
    - Online / Offline triplet mining



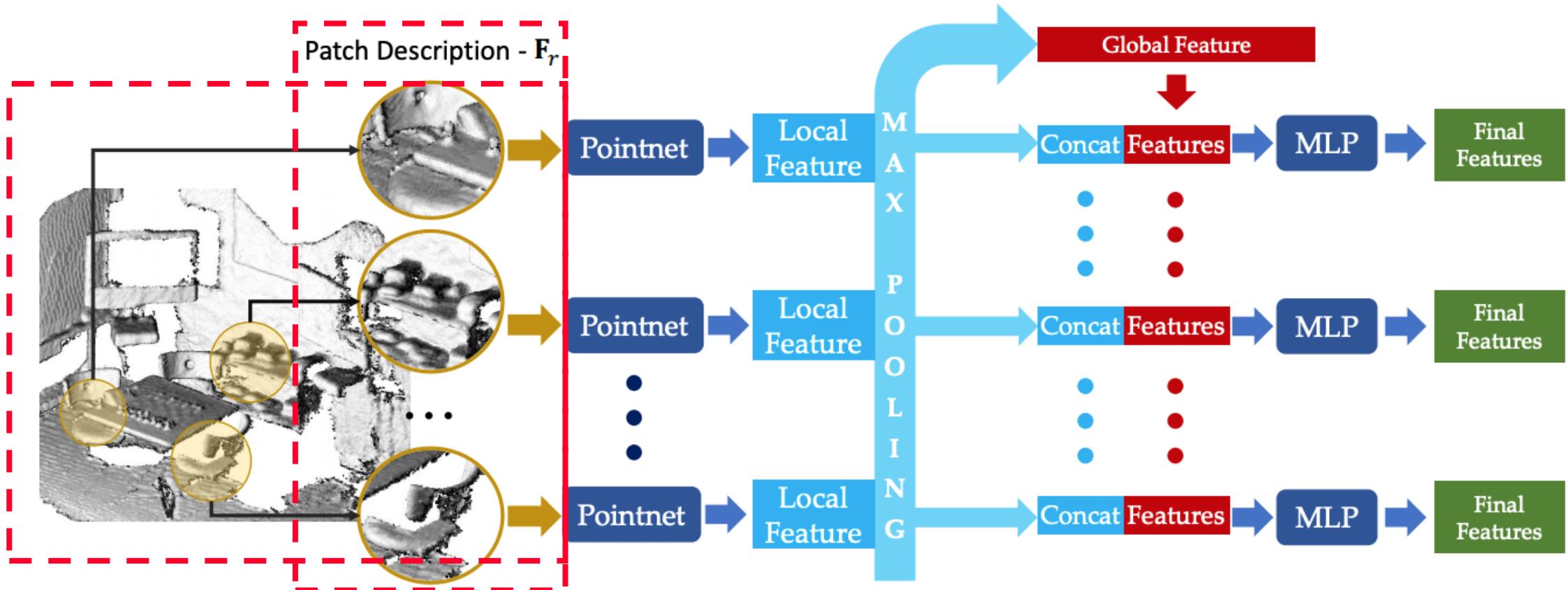
3DMatch - Architecture



The Perfect Match - Local Reference Frame



## How to encode a patch for PointNet



Consider all patches in one frame at the same time.



- Simplest way
  - For query point  $x_r \in \mathbb{R}^3$ , find its neighbors  $\Omega$  by kNN/RadiusNN
  - Stack the  $M$  points into  $M \times 3$  matrix
- Generally more information is better
  - Surface normal vectors improve classification/segmentation (Lecture 5)
- What else?
  - Pair wise features like the PFH?



- For a pair of points  $x_1, x_2$
- The PPF is

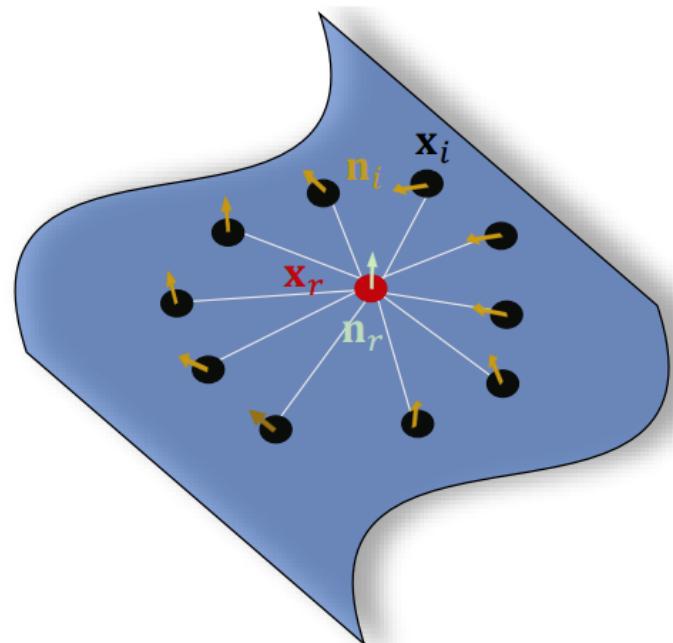
$$\psi_{12} = (\|\mathbf{d}\|_2, \angle(\mathbf{n}_1, \mathbf{d}), \angle(\mathbf{n}_2, \mathbf{d}), \angle(\mathbf{n}_1, \mathbf{n}_2))$$

$$\angle(\mathbf{v}_1, \mathbf{v}_2) = \text{atan2} (\|\mathbf{v}_1 \times \mathbf{v}_2\|, \mathbf{v}_1 \cdot \mathbf{v}_2)$$

- For a patch around  $x_r$ ,
  - Compute  $\psi_{ri}, \forall i \in \Omega$
  - The PPF for the patch is

$$\mathbf{F}_r = \{\mathbf{x}_r, \mathbf{n}_r, \mathbf{x}_i, \dots, \mathbf{n}_i, \dots, \psi_{ri}, \dots\}$$

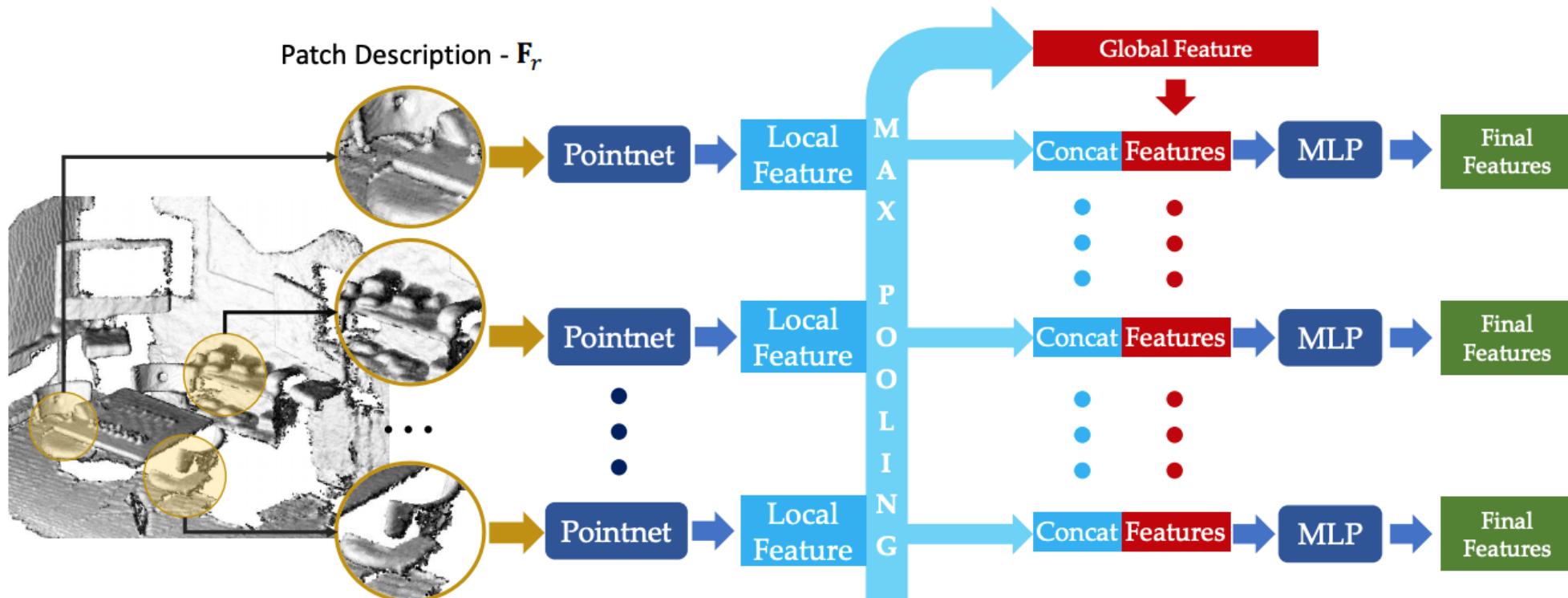
Local Patch Encoding



$$\mathbf{F}_r = \begin{bmatrix} \mathbf{x}_0 \\ \dots \\ \mathbf{x}_k \\ \mathbf{n}_0 \\ \dots \\ \mathbf{n}_k \\ ppf(\mathbf{x}_r, \mathbf{x}_0, \mathbf{n}_r, \mathbf{n}_0) \\ ppf(\mathbf{x}_r, \mathbf{x}_1, \mathbf{n}_r, \mathbf{n}_1) \\ \dots \\ ppf(\mathbf{x}_r, \mathbf{x}_k, \mathbf{n}_r, \mathbf{n}_k) \end{bmatrix}$$

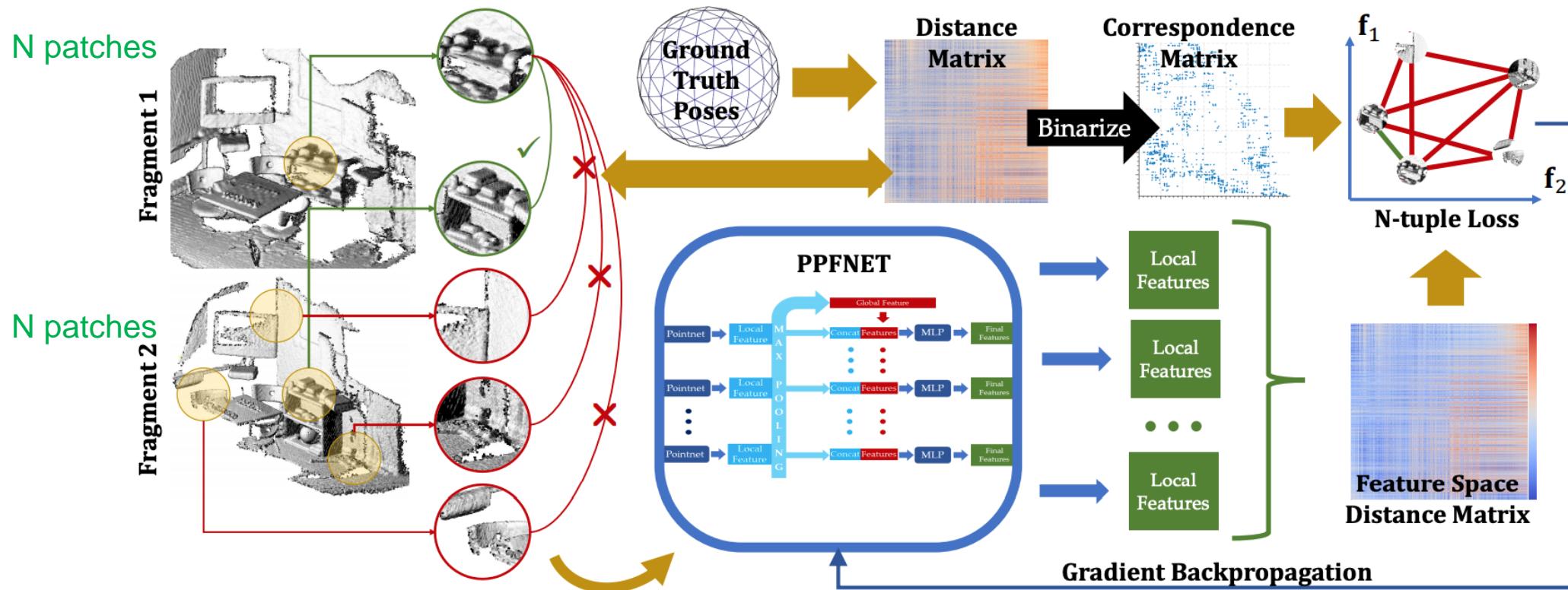


- Local Feature: feed  $F_r$  into a shared PointNet.
- Global Feature: maxpool of all local features in the same frame.
- Final Feature: concatenation of Local Feature and Global Feature + MLP





- Input: A pair of frames (fragments) with **known ground truth pose**
- Positive pairs are trivial – Ground truth transformation is given at training
- Loss function – How to make use of “global” information?





- Contrastive Loss

$$L = \frac{1}{N} \sum_{n=1}^N y_{ij} d_{ij}^2 + (1 - y_{ij}) \max(\tau - d_{ij}, 0)^2$$

- Considers **2 samples** at a time. A+N / A+P

- Triplet Loss

$$L = \sum_{i=1}^N L_i = \sum_{i=1}^N \max(d_i(a, p) - d_i(a, n) + \gamma, 0)$$

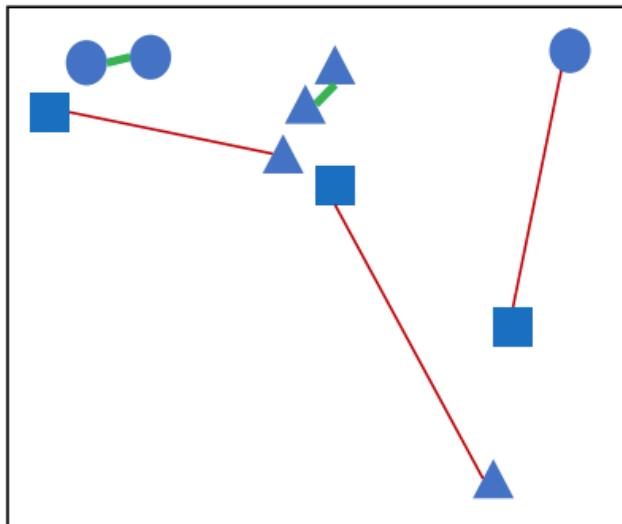
- Considers **3 samples** at a time. A + N + P

- Can we consider **N samples** at a time?

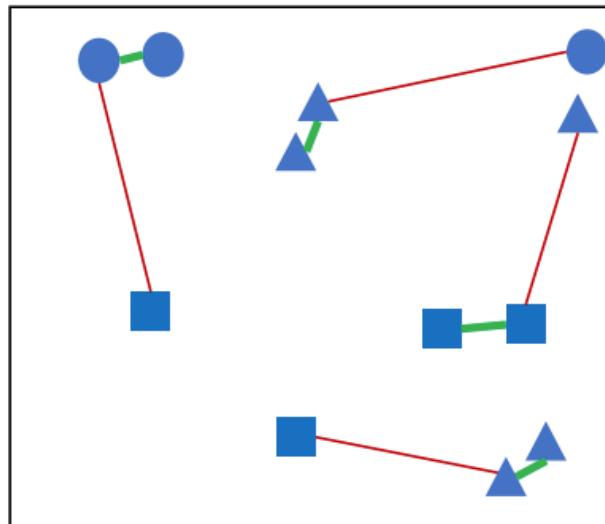
- PPFNet considers two frames at a time
- N samples at each frame



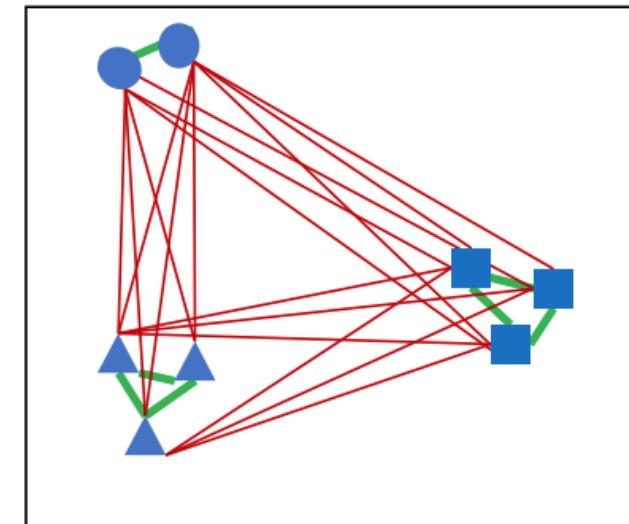
- N-tuple Loss
  - N keypoints in each of the two frames
  - N to N association → N tuples
  - Similar samples stay together, dis-similar samples stay away



**(a)** Pair Samples



**(b)** Triplet Samples



**(c)** N-Tuple Configuration



1. There are  $N$  patches in each frame
    - Frame 1: keypoint locations  $x_i$
    - Frame 2: keypoint locations  $y_i$
    - Transformation matrix between frames:  $T$
  2. Construct the **Euclidean space** patch distance matrix
    - Distance between  $N$  keypoints in frame1 to  $N$  keypoints in frame2
  3. Keypoint distance matrix  $\rightarrow$  Corresponding matrix  $M \in \mathbb{R}^{N \times N}$

$$m_{ij} = \mathbb{1}(\|\mathbf{x}_i - \mathbf{T}\mathbf{y}_j\|_2 < \tau)$$

Indicator function

Threshold to determine positive pairs



4. Compute **feature space** distance matrix  $\mathbf{D} \in \mathbb{R}^{N \times N}$  between descriptors  $f(\mathbf{x}_i), f(\mathbf{y}_j)$ .  $f(\cdot)$  is the PPFNet.

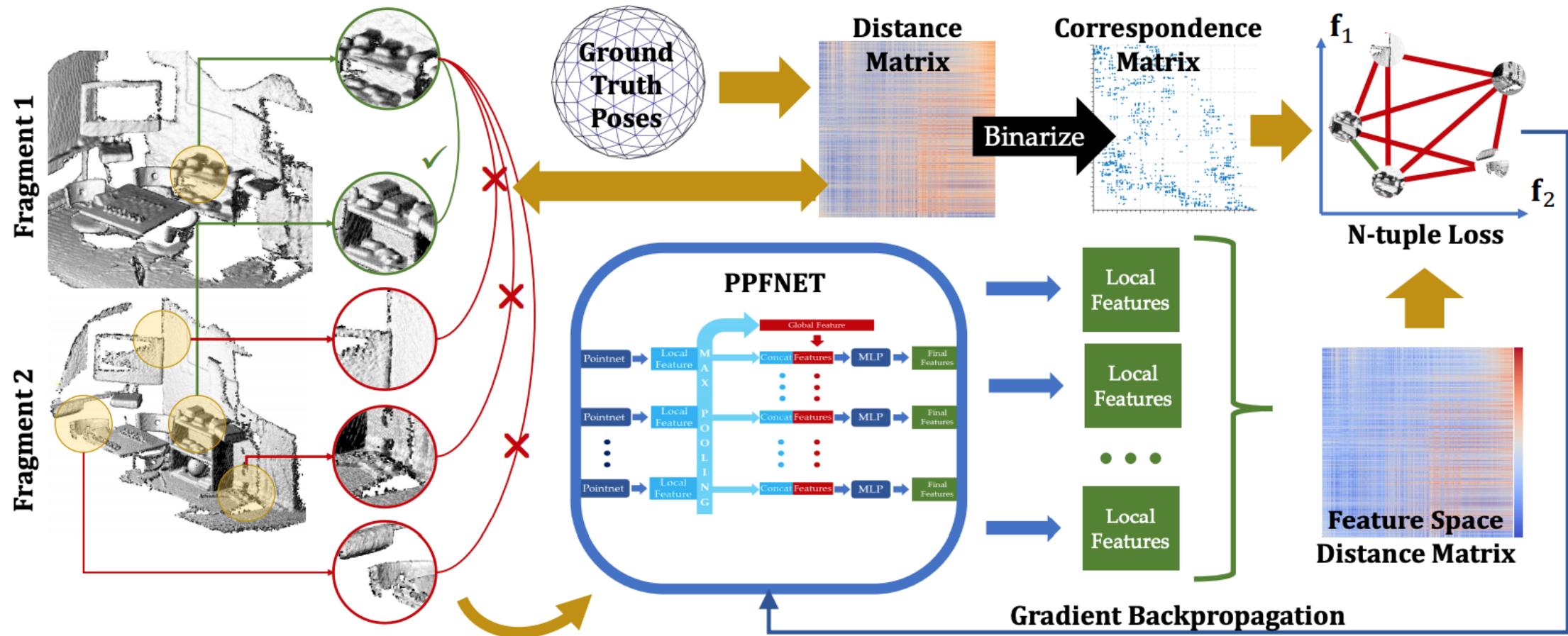
$$d_{ij} = \|f(\mathbf{x}_i) - f(\mathbf{y}_j)\|_2$$

5. The N-tuple loss is given by

$$L = \sum^* \left( \frac{\mathbf{M} \circ \mathbf{D}}{\|\mathbf{M}\|_2^2} + \alpha \frac{\max(\theta - (1 - \mathbf{M}) \circ \mathbf{D}, 0)}{N^2 - \|\mathbf{M}\|_2^2} \right)$$

Annotations pointing to parts of the equation:

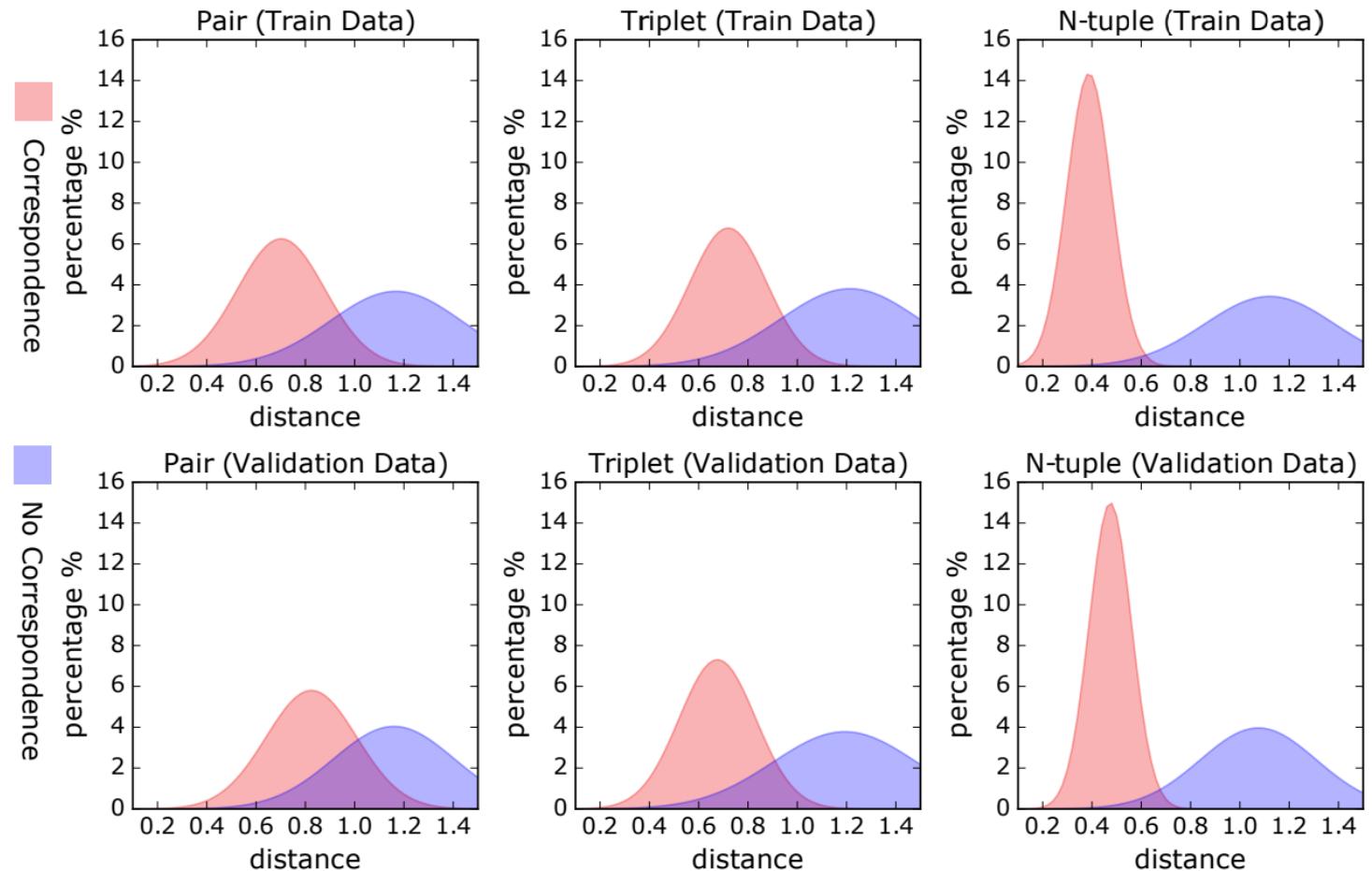
- Positive descriptor distance: Points to  $\frac{\mathbf{M} \circ \mathbf{D}}{\|\mathbf{M}\|_2^2}$ .
- Negative descriptor distance: Points to  $\frac{\max(\theta - (1 - \mathbf{M}) \circ \mathbf{D}, 0)}{N^2 - \|\mathbf{M}\|_2^2}$ .
- Number of 1 (# of positive matches): Points to the term  $\mathbf{M} \circ \mathbf{D}$ .
- Hyper-parameter that balance the weight between positive and negative: Points to the term  $\alpha$ .
- Margin of typical contrastive loss: Points to the term  $\max(\theta - (1 - \mathbf{M}) \circ \mathbf{D}, 0)$ .
- Element-wise multiplication: Points to the term  $\mathbf{M} \circ \mathbf{D}$ .





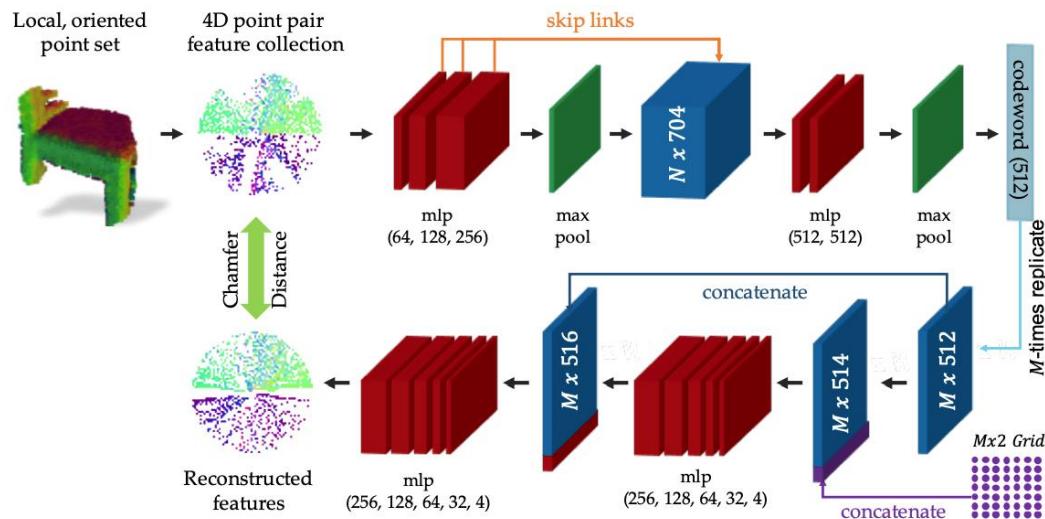
## PPFNet – N-tuple Loss

- Pink: descriptor distance between positive pairs
- Purple: descriptor distance between negative pairs
- Column 1:  
Contrastive loss
- Column 2:  
Triplet loss
- Column 3:  
N-tuple loss

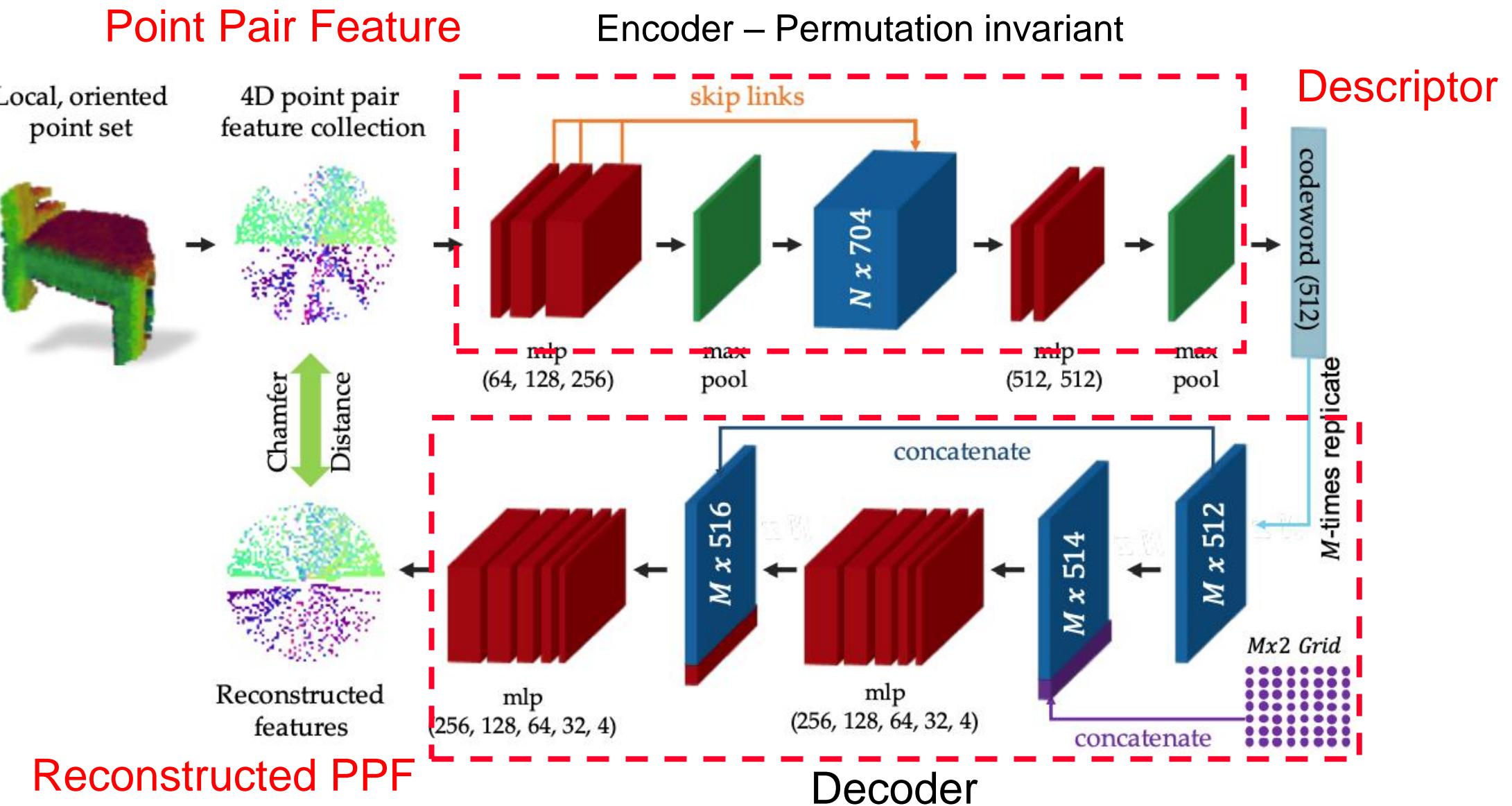




- Common points about 3DMatch, PerfectMatch, PPFNet
  - Relies on loss that pulls positive pairs & push negative pairs
  - Requires ground truth label of positive / negative, via ground truth pose
- No contrastive / triple loss? Unsupervised? – PPF-FoldNet



Source: PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors, Haowen Deng, et.al.



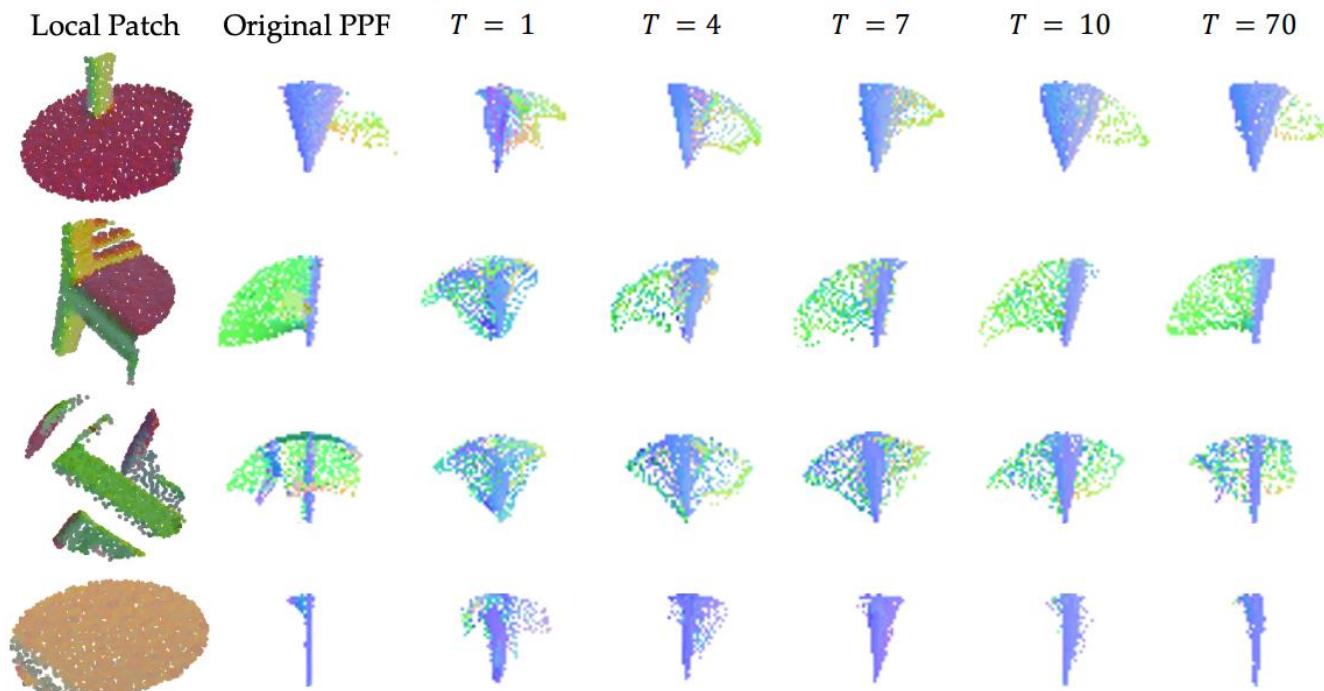


- Inference – Green. Training – Blue.
1. Convert a patch into PPF features  $F = \{\psi_{r1}, \dots, \psi_{ri}, \dots \psi_{rn}\} \in \mathbb{R}^{n \times 4}$   
 $\psi_{12} = (\|d\|_2, \angle(n_1, d), \angle(n_2, d), \angle(n_1, n_2))$  -- **Rotation Invariant**
  2. Get the codeword (descriptor, length 512) with Encoder
  3. Get the reconstructed PPF features  $\hat{F} \in \mathbb{R}^{n \times 4}$  with Decoder
  4. Compute Chamfer distance between  $F, \hat{F}$



$$d(\mathbf{F}, \hat{\mathbf{F}}) = \max \left\{ \frac{1}{|\mathbf{F}|} \sum_{\mathbf{f} \in \mathbf{F}} \min_{\hat{\mathbf{f}} \in \hat{\mathbf{F}}} \|\mathbf{f} - \hat{\mathbf{f}}\|_2, \frac{1}{|\hat{\mathbf{F}}|} \sum_{\hat{\mathbf{f}} \in \hat{\mathbf{F}}} \min_{\mathbf{f} \in \mathbf{F}} \|\mathbf{f} - \hat{\mathbf{f}}\|_2 \right\}$$

Original & reconstructed PPF in different training epochs



### Registration Recall on 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1937	0.1779	0.3063	0.5751	0.4605	<b>0.8972</b>	0.5949	0.7352	0.7866
Home 1	0.3974	0.3718	0.5833	0.7372	0.6154	0.5577	0.7179	0.7564	<b>0.7628</b>
Home 2	0.3654	0.3365	0.4663	<b>0.7067</b>	0.5625	0.5913	0.6058	0.6250	0.6154
Hotel 1	0.1814	0.2080	0.2611	0.5708	0.4469	0.5796	0.6549	0.6593	<b>0.6814</b>
Hotel 2	0.2019	0.2212	0.3269	0.4423	0.3846	0.5769	0.4231	0.6058	<b>0.7115</b>
Hotel 3	0.3148	0.3889	0.5000	0.6296	0.5926	0.6111	0.6111	0.8889	<b>0.9444</b>
Study	0.0548	0.0719	0.1541	0.5616	0.4075	0.5342	<b>0.7123</b>	0.5753	0.6199
MIT Lab	0.1039	0.1299	0.2727	0.5455	0.3506	<b>0.6364</b>	0.5844	0.5974	0.6234
Average	0.2267	0.2382	0.3589	0.5961	0.4776	0.6231	0.6130	0.6804	<b>0.7182</b>

### Registration Recall on **Rotated** 3DMatch Dataset

	Spin Image [16]	SHOT [35]	FPFH [34]	3DMatch [51]	CGF [18]	PPFNet [8]	FoldNet [48]	Ours	Ours-5K
Kitchen	0.1779	0.1779	0.2905	0.0040	0.4466	0.0020	0.0178	0.7352	<b>0.7885</b>
Home 1	0.4487	0.3526	0.5897	0.0128	0.6667	0.0000	0.0321	0.7692	<b>0.7821</b>
Home 2	0.3413	0.3365	0.4712	0.0337	0.5288	0.0144	0.0337	0.6202	<b>0.6442</b>
Hotel 1	0.1814	0.2168	0.3009	0.0044	0.4425	0.0044	0.0133	0.6637	<b>0.6770</b>
Hotel 2	0.1731	0.2404	0.2981	0.0000	0.4423	0.0000	0.0096	0.6058	<b>0.6923</b>
Hotel 3	0.3148	0.3333	0.5185	0.0096	0.6296	0.0000	0.0370	0.9259	<b>0.9630</b>
Study	0.0582	0.0822	0.1575	0.0000	0.4178	0.0000	0.0171	0.5616	<b>0.6267</b>
MIT Lab	0.1169	0.1299	0.2857	0.0260	0.4156	0.0000	0.0260	0.6104	<b>0.6753</b>
Average	0.2265	0.2337	0.3640	0.0113	0.4987	0.0026	0.0233	0.6865	<b>0.7311</b>



	Representation	Loss Function	Rotation Handling
3DMatch	Voxel grid	Contrastive Loss	No
PerfectMatch	Voxel grid	Triplet Loss	Local Reference Frame
PPFNet	Coordinates + surface normal+ PPF	N-tuple Loss	No
PPF-FoldNet	PPF	Reconstruction Chamfer	Rotation invariant



- Handcrafted
  - PFH, FPFH, SHOT
  - Advantages
    - Captures local surface variations
    - Invariant to 6D pose
  - Disadvantages:
    - Sensitive to surface normal estimation
    - Sensitive to noise / occlusion / sparsity.
- Deep Learning
  - 3DMatch, PerfectMatch, PPFNet, PPF-FoldNet
  - Advantages:
    - More distinctive, better performance
    - Robust to noise / occlusion / sparsity
  - Disadvantages:
    - Requires GPU
    - Requires training
    - Generalization



## Homework

- Implement your own ISS keypoint detection.
- Implement feature descriptors FPFH, SHOT
  - You may call PCL library via python binding **OR** implement your own version.
  - Example of python binding of PCL: <https://github.com/lijx10/PCLKeypoints>
  - Test with ModelNet40 to ensure correctness.
    - For example, find a symmetric table, compute descriptor around the end-point of each leg. Check whether the descriptors are similar. **Submit a report**.
    - The correctness will be reflected in Lecture 9 homework.
- Register two point clouds by feature detection and description
  - Homework of Lecture 9.