第230回CVIM研究発表会 チュートリアル
深層学習を用いた三次元点群処理入門

早稲田大学 / OSX
千葉 直也

本日の内容

- 三次元点群に深層学習を適用する際の
  - キーとなるアイデア
  - 典型的なネットワーク構成
  - 点群畳み込み
  - 点群（・三次元形状）の出力

  について解説
- アプリケーション例を紹介
  - 特に点群位置合わせに関して

アウトライン

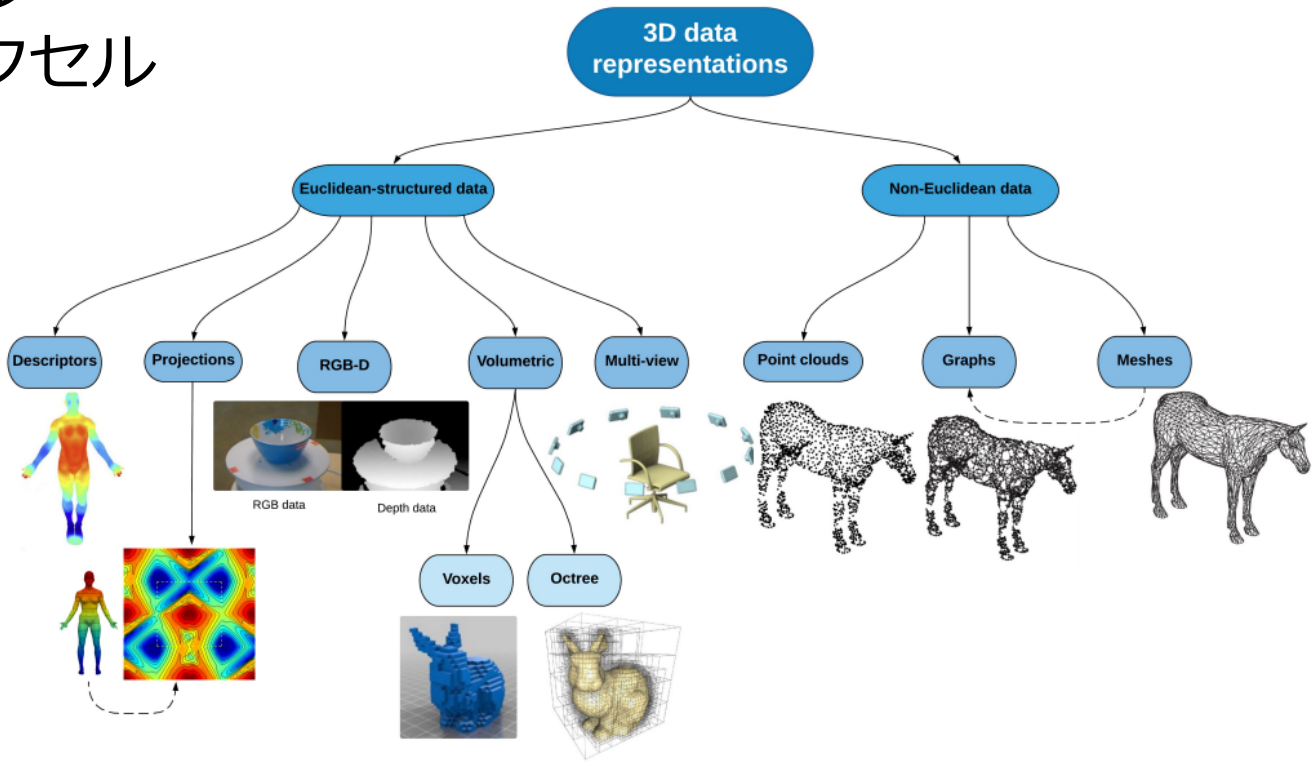
- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- 点群を扱うニューラルネットワーク
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

アウトライン

- **三次元形状のデータ形式と計測**
  - **さまざまなデータ形式**
  - **三次元計測**
- 点群を扱うニューラルネットワーク
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

三次元形状を記述するデータ形式

- ボクセル
  - 密なボクセル
  - スパースボクセル
- メッシュ
- 多視点画像
- 深度画像
- 点群
  - 片面点群
  - 全周点群



様々な形式が存在，今回は三次元点群に着目

E. Ahmed+. A survey on Deep Learning Advances on Different 3D Data Representations. arXiv, 2018.

三次元データの形式（1/2）

- **ボクセル**
  - 画像に近いアイデアで記述
  - ○: シンプルなデータ形式．3D CNNで処理しやすい
  - △: 解像度に応じて急速にデータ量が大きくなるため，扱える解像度が低い．スパースボクセルで対応
- **メッシュ**
  - 頂点（・辺）・面によって三次元形状を記述
  - ○: 省データで高解像度なデータを表せる．レンダリングとの相性が良い
  - △: 頂点・辺・面をうまく扱うのが難しい．辺や面はセンサから得られない

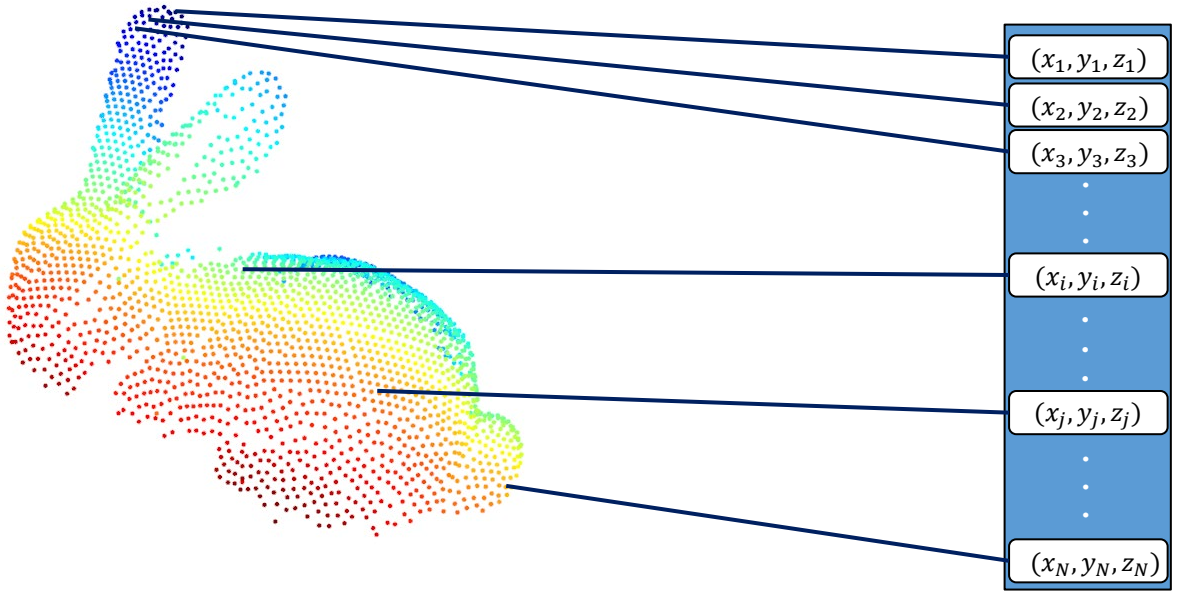
三次元データの形式（2/2）

- **深度画像**
  - 2D画像の各画素に紐付いた奥行きを記述
  - ○: シンプルでセンサとの相性も良い.
    2D CNNのテクニックが使いやすい
  - △: 全周形状が扱いにくい
- **多視点画像**
  - 多点のカメラ画像によって一つの三次元シーンを記述
  - ○: 全周が扱える．2D CNNのテクニックが使いやすい
  - △: 実環境では計測しにくい.
    実スケール性を活用しにくい

三次元点群とは

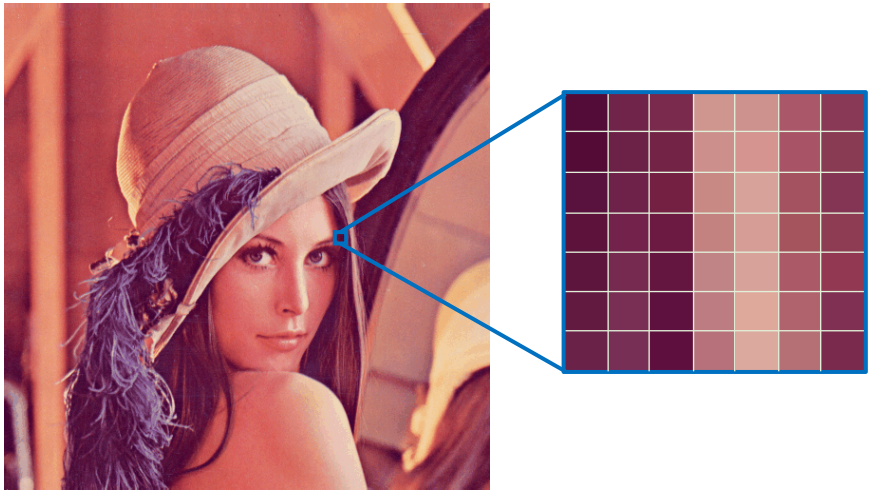
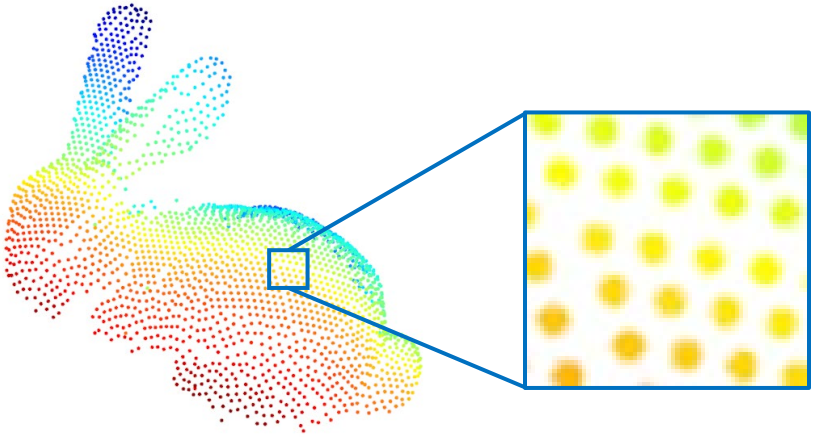
三次元点 (x, y, z) の集合として表現された
三次元形状の記述方法

データによってはカラー，反射強度などの付加情報が
点ごとに与えられる



三次元点群の表す形状　　三次元点群データ

二次元画像と三次元点群の違い

二次元画像		**三次元点群**
		
二次元上のピクセル	構成要素	**三次元上の点**
密 （画像上に画素が詰まっている）	粗密	**疎** **（表面を表す点のみ）**
あり	構成要素の順序	**なし**

3Dデータの種類

（主観ですが）各種3Dデータの特徴をまとめてみました

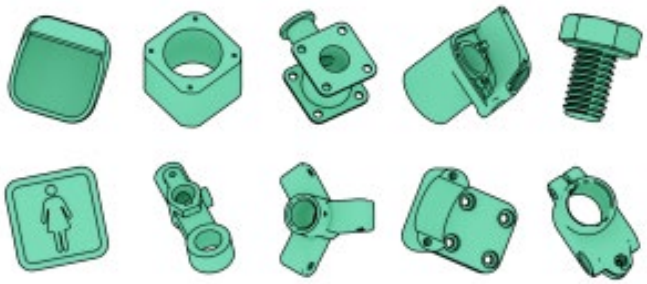
点群は実スケール性がありシンプルな形式

	実スケール	扱いやすさ	メモリ 空間効率	CNNとの 相性	計測 しやすさ	別視点の レンダリング
ボクセル	◎	◎	×	◎	△	◎
スパース ボクセル	◎	○	○	○	△	◎
メッシュ	◎	△	◎	△	△	◎
多視点画像	△	○	△	◎	○～◎	△～○
深度画像	○	◎	○	◎	◎	△
点群	◎	○	◎	○	◎	△
全周点群	◎	○	◎	○	△	○

3Dデータの取得

- 基本的には**三次元計測**
  - 次以降のスライドで簡単に紹介
  - 全周形状・360度形状の計測には貼り合わせが必要
- MVS, SfM, SLAM等による多視点計測データ
  - 計測原理は三角測量．NeRFに期待
- **CADや3Dモデル**など
  - 工業用途・CG等で利用されているデータ
  - 一から計算機の中で製作される

MVSで再構成された3Dデータ（COLMAP）



CADデータ（ABC Dataset）

三次元センサの種類

ステレオ法

- 2つの光学系の間の視差から奥行きを計算
- ステレオカメラシステムを用いる手法（パッシブステレオ法）と，プロジェクタ・カメラシステムを用いる手法（アクティブステレオ法）に大別

Time-of-Flight (ToF)

- 光を投影し，反射した光線が返ってくるまでの時間から奥行きを計算
- 時間差を用いる方式（direct ToF）と位相差を用いる方式（indirect ToF）に大別

三次元計測の原理（ステレオ）

２つの光学系での**ステレオ視差**で奥行きを求める

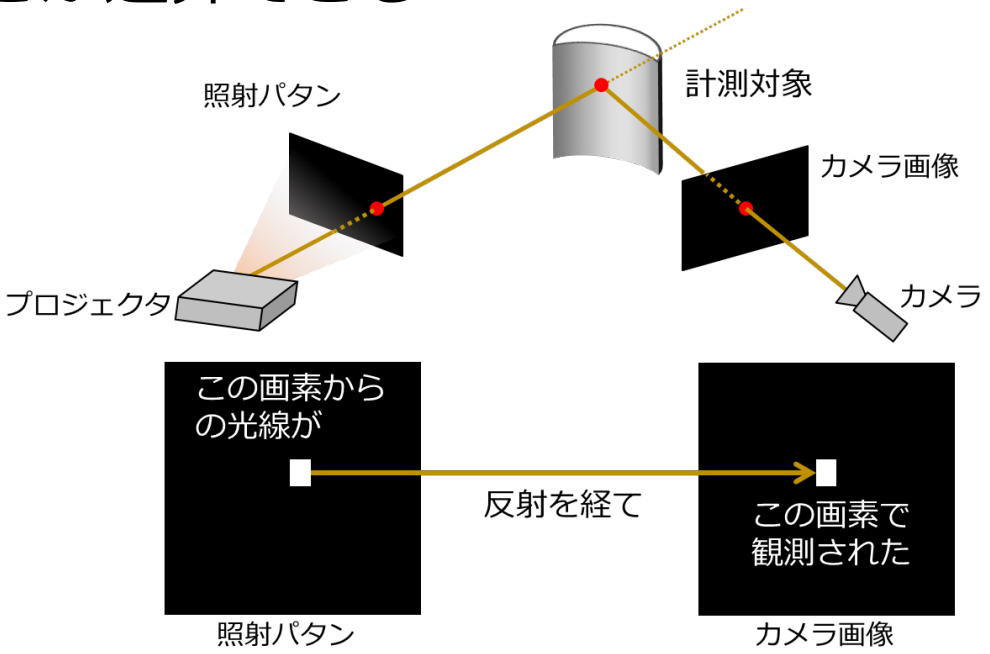
アクティブステレオ法：
プロジェクタ・カメラシステムでの視差をもとに奥行きを推定.
計測対象の表面テクスチャによらず計測できるため,
工業用途で普及

パッシブステレオ法：
２つのカメラ間の視差をもとに奥行きを推定.
計測対象にテクスチャがない場合（白い壁など）に対応するため，ランダムなパタンを投影して視差を計算しやすくする例が多い.
RealSense Dシリーズなど

三次元計測の原理（アクティブステレオ）

画素の対応関係がわかる→奥行きがわかる

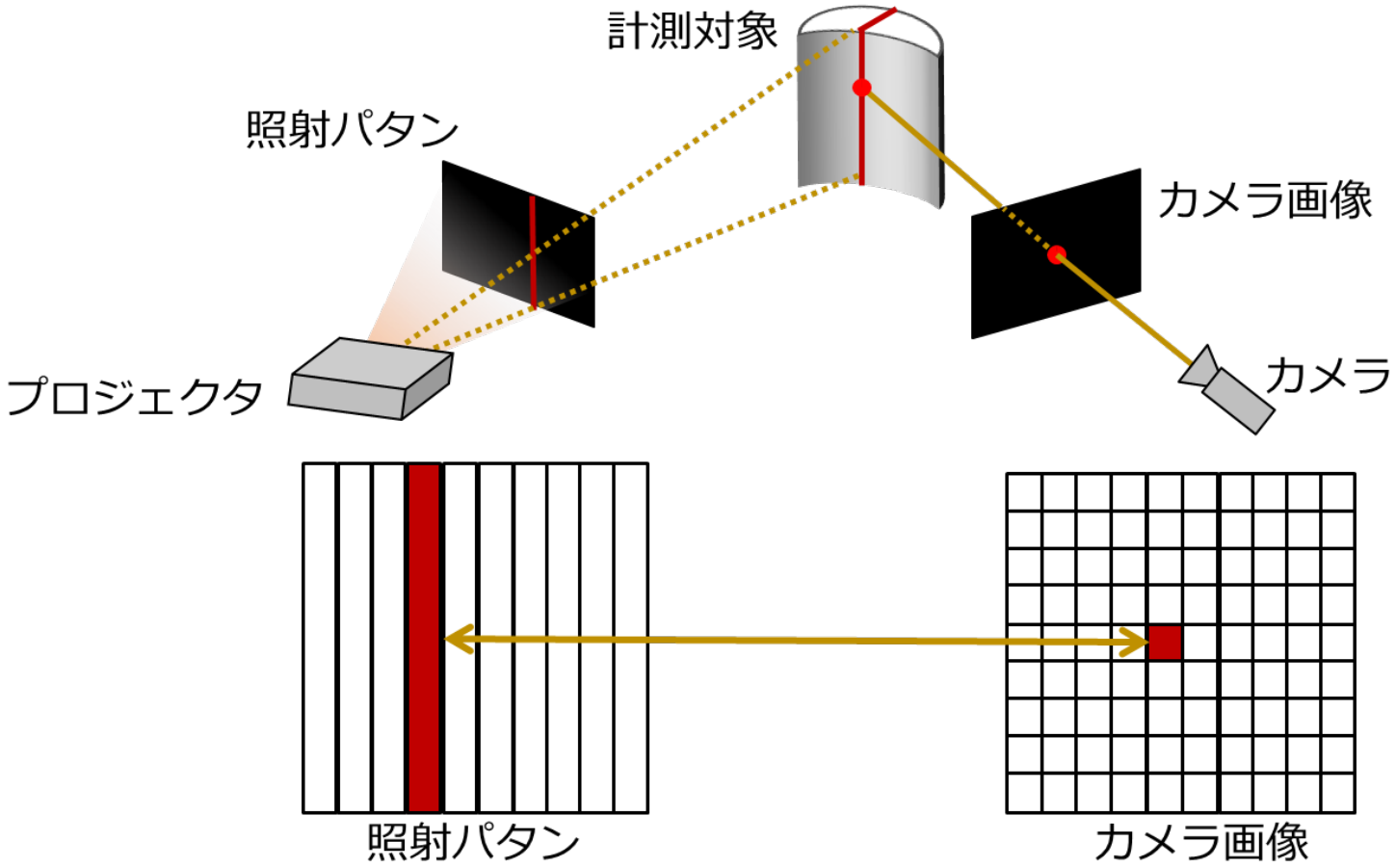
- 内部パラメータ・外部パラメータが既知のステレオ光学系を想定
- あるプロジェクタ画素から照射した光線が，どのカメラ画素で観測されたかがわかれば奥行きが逆算できる



三次元計測の原理（アクティブステレオ）

画素の対応関係がわかる→奥行きがわかる

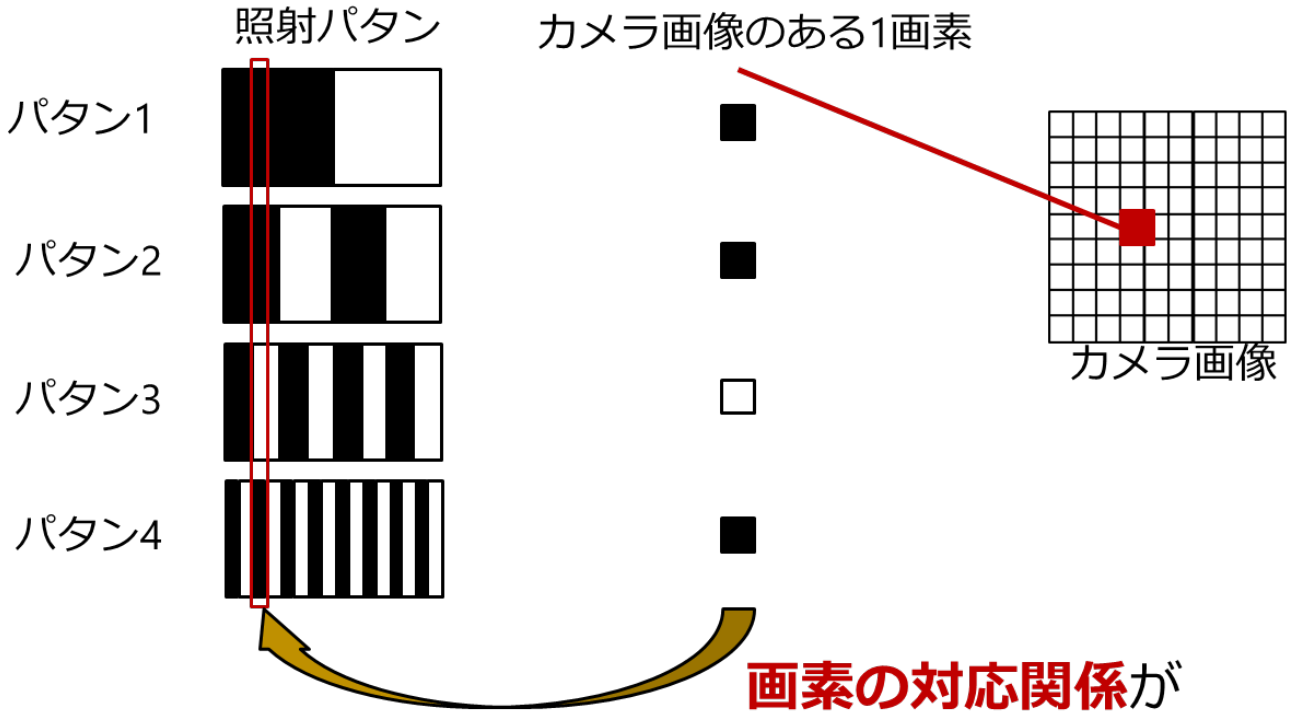
　実際には画素の対応ではなく，行or列との対応だけで十分



三次元計測の原理（アクティブステレオ）

空間コード化法

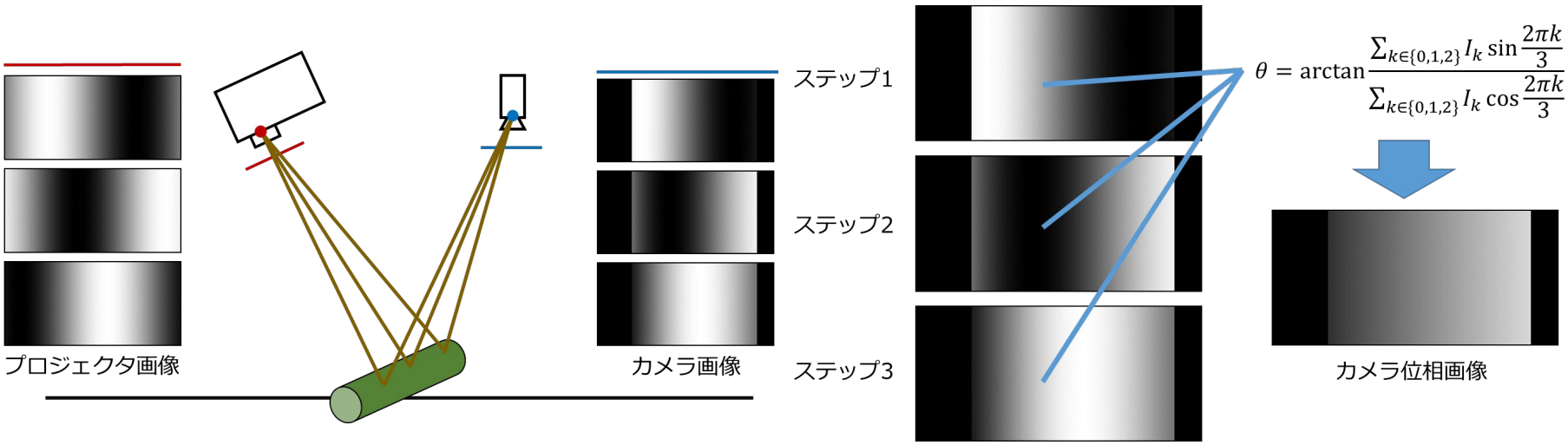
プロジェクタの各画素に対応するコードを割り当て，
カメラの各画素で
観測された光線から対応するプロジェクタ画素を求める



三次元計測の原理（アクティブステレオ）

位相シフト法

プロジェクタの各画素に位相値を割り当て，
サイン波パタンを照射し計測．
カメラの各画素について位相値を計算して
位相値で対応を求める



三次元計測の原理（ToF）

光の速さ×かかった時間で奥行きを求める

直接法（direct ToF）
時間を直接計算する.
外乱光に強く遠距離の計測・室外環境での
計測に向くが，比較的高価.
Velodyne LiDARなど

間接法（indirect ToF）
周期的な光を照射し，反射光との位相差から
距離を逆算．比較的安価だが外乱光に弱い傾向にある.
Azure Kinectなど

アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - **点群を扱う難しさ**
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

三次元点群を取り扱う難しさ

要因: **集合データであること**

- **順不同**である（後述）ため，それに対応したネットワーク構造が必要
- **非グリッド**なので近傍が自明ではなく，畳み込みにも工夫が必要

（余談: グラフNNとの比較）

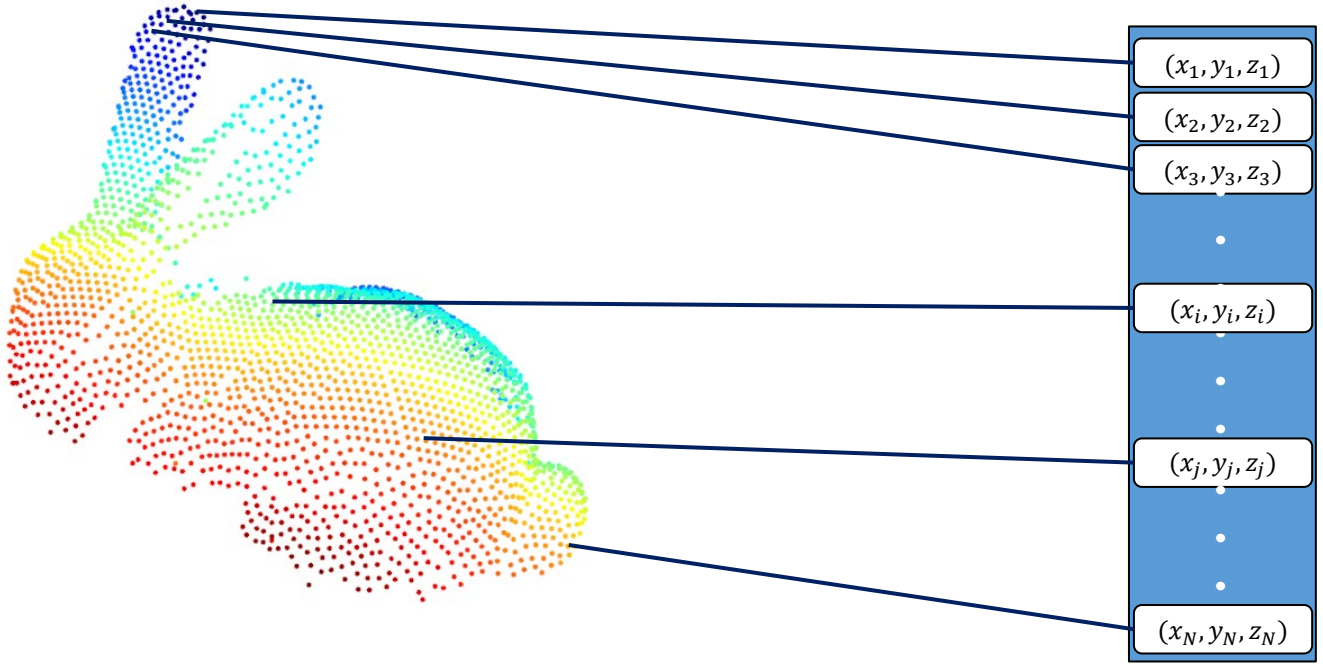
- グラフNNでも同様の課題があり，ほぼ同じアプローチが取られている
- 点群の場合には三次元空間に張り付いている＆接続関係は入力されない，という違いがある

三次元点群を取り扱う難しさ

順不同なデータ構造

点群は点の集合＝データの順序が変わっても同じ形状

→ 点の順序に不変であることが必要



三次元点群の表す形状

三次元点群データ

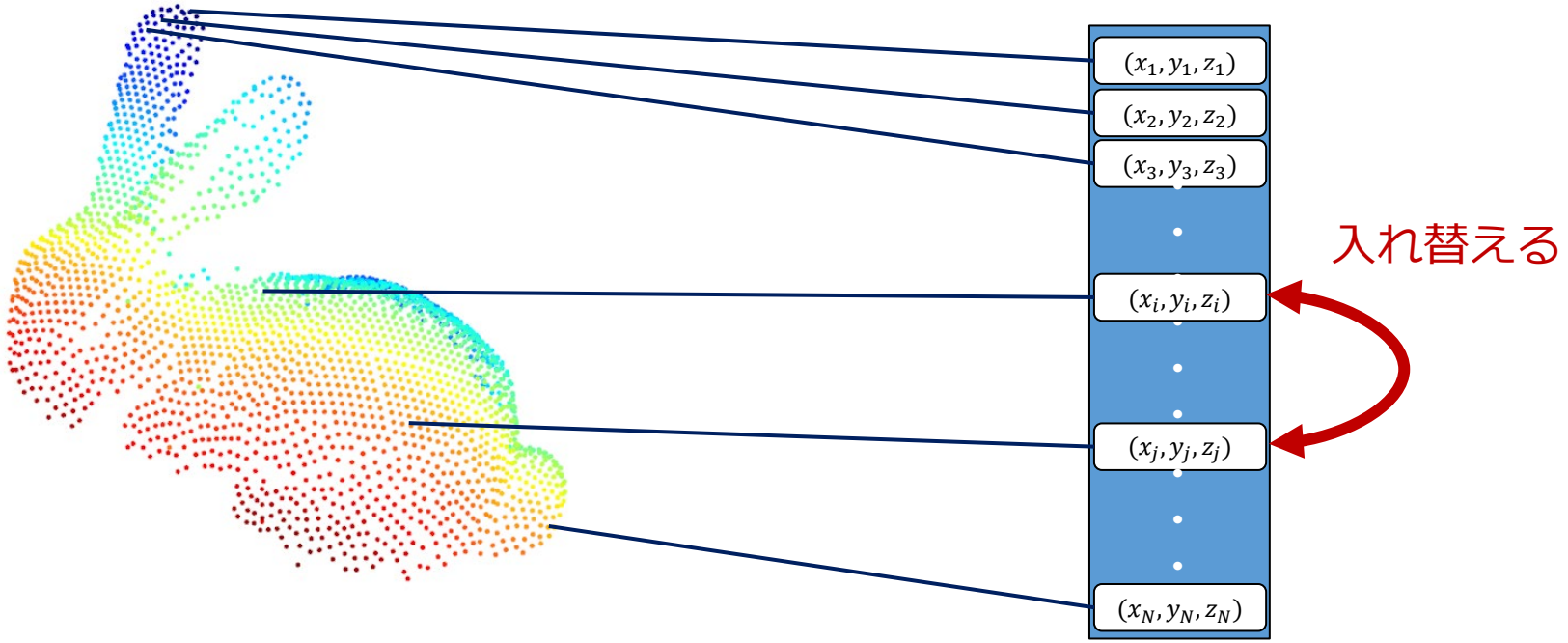
あくまで便宜上一列にして扱う

三次元点群を取り扱う難しさ

順不同なデータ構造

点群は点の集合＝データの順序が変わっても同じ形状

→ 点の順序に不変であることが必要



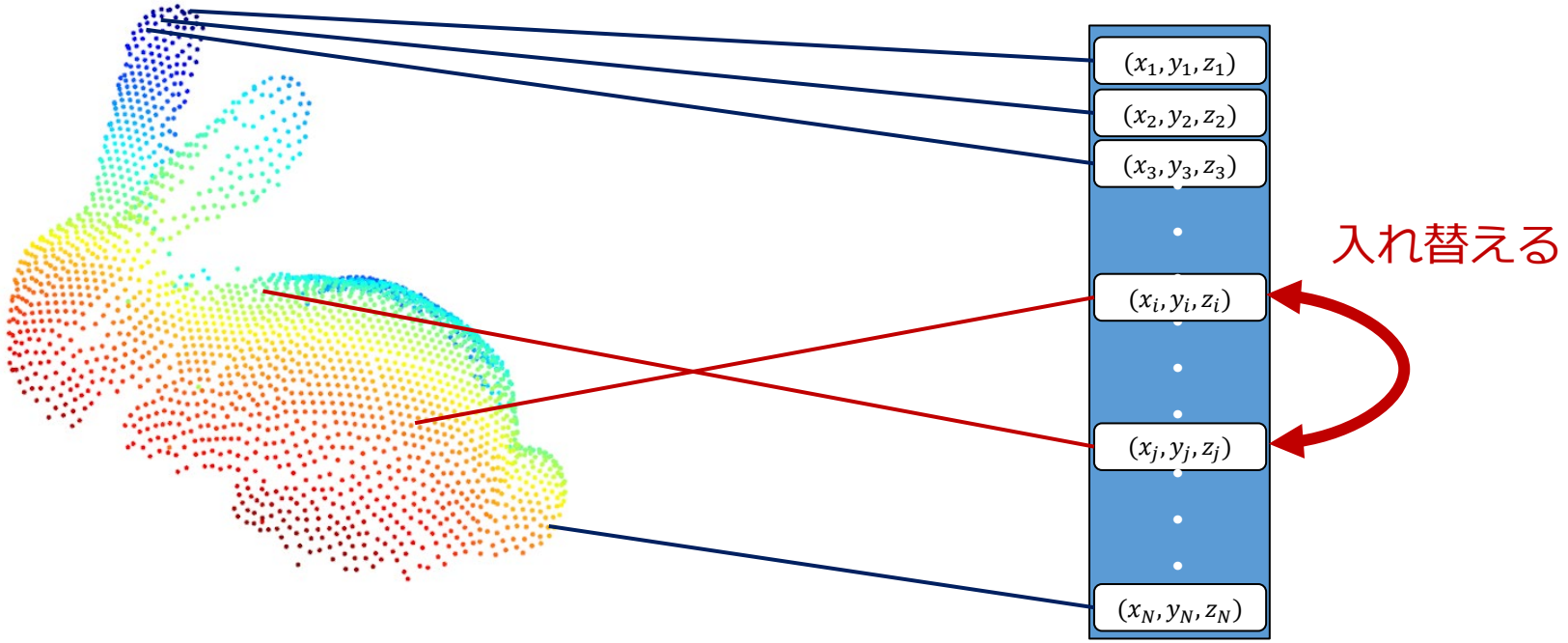
三次元点群の表す形状

三次元点群データ

三次元点群を取り扱う難しさ

順不同なデータ構造

点群は点の集合＝データの順序が変わっても同じ形状
→ 点の順序に不変であることが必要



三次元点群の表す形状

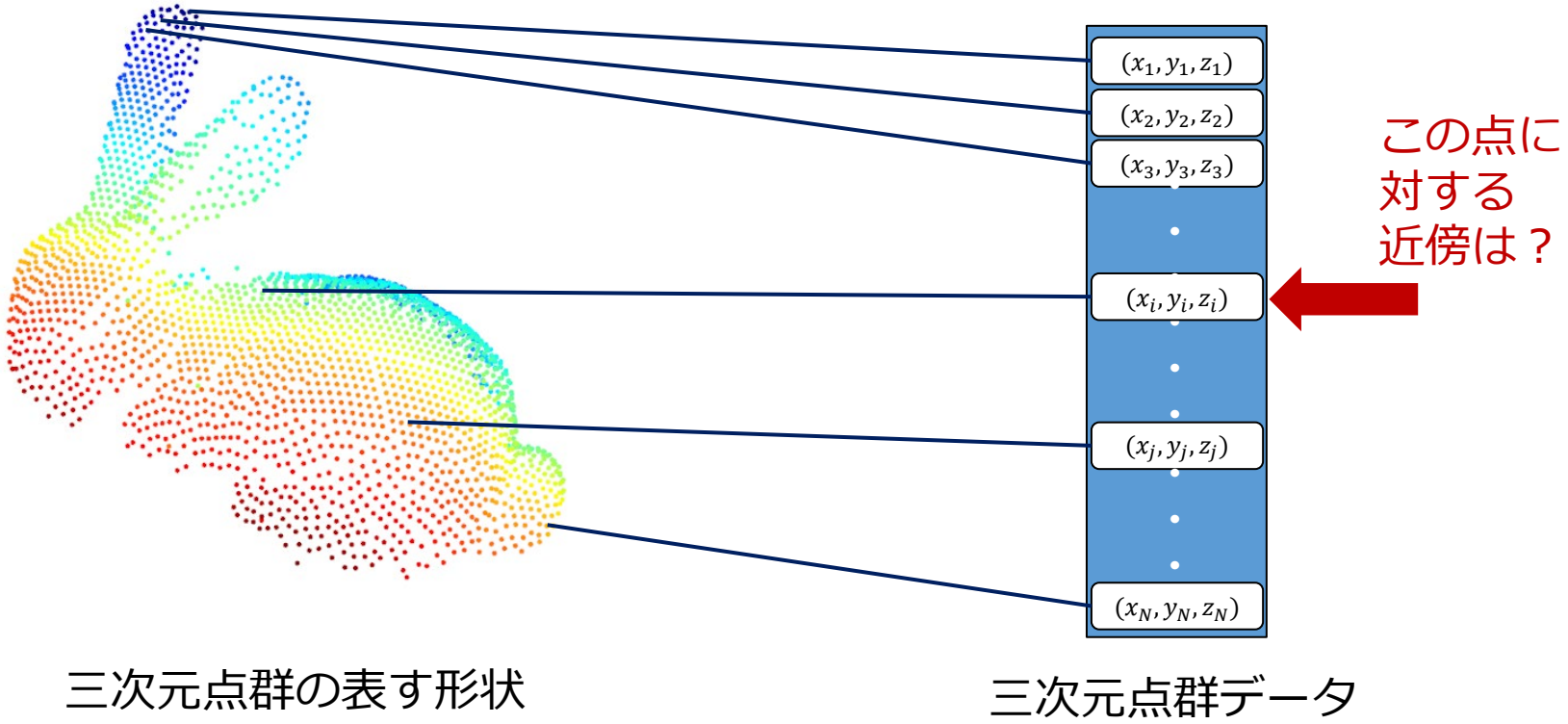
三次元点群データ

順序を入れ替えても点群としての形状は全く同じ

三次元点群を取り扱う難しさ

非グリッド＝非自明な隣接関係

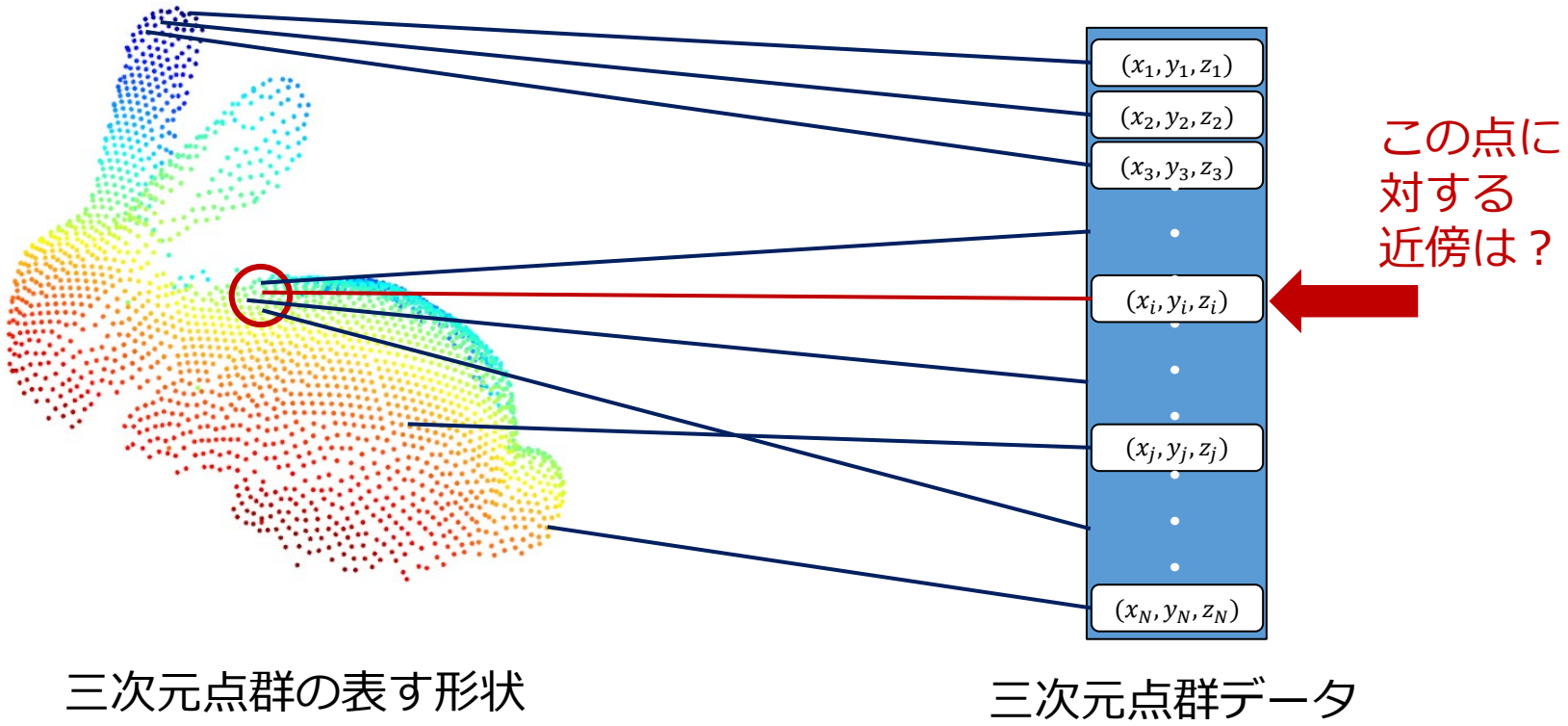
畳み込みによる局所特徴の抽出には工夫が必要



三次元点群を取り扱う難しさ

非グリッド＝非自明な隣接関係

畳み込みによる局所特徴の抽出には工夫が必要



データ上の隣接関係は当然関係ない

アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - **典型的なタスク・ネットワーク構成**
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

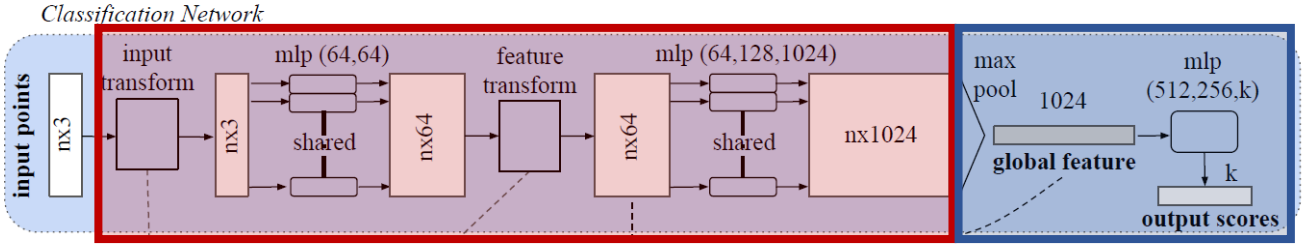
典型的なタスク例の分類

- **点群全体について推定**
  - クラス分類
  - 位置・姿勢推定
  - 変形推定
  - 全周点群補完
  - 点群（など，3D形状）を出力
- **インスタンスについて推定**
  - 物体検出
- **各点について推定**
  - セマンティックセグメンテーション
  - デノイジング

点群を扱うNNの典型的な構成

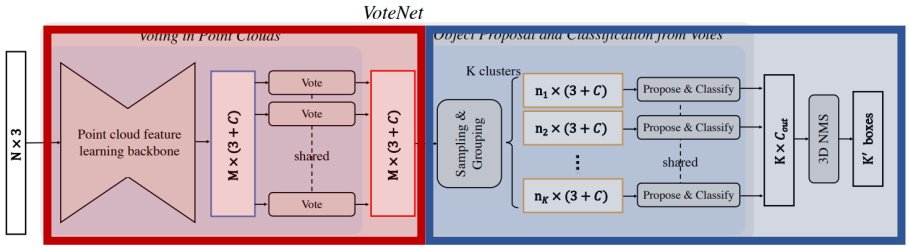
- **点群畳み込みで特徴抽出→タスクに合わせて出力**
  - 2D CNNと同様にBackboneと呼ばれるようになってきた
  - 徐々に密度を下げる構造，ボトルネック構造，スキップ接続等もよく用いられる
- 出力部分（いわゆるHead部分）：タスクに合わせて設計，後処理を含むことも

PointNet
クラス分類



Backbone　　Head

VoteNet
物体検出



Backbone　　Head

アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - **PointNetの紹介**
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

PointNetの紹介

Symmetric Functionを使うアイデア

PointNet [Qi+, CVPR2017], Deep Sets [Zaheer+, NIPS2017]

ポイント1:

点ごとの変換であれば順序に依存しない

ポイント2:

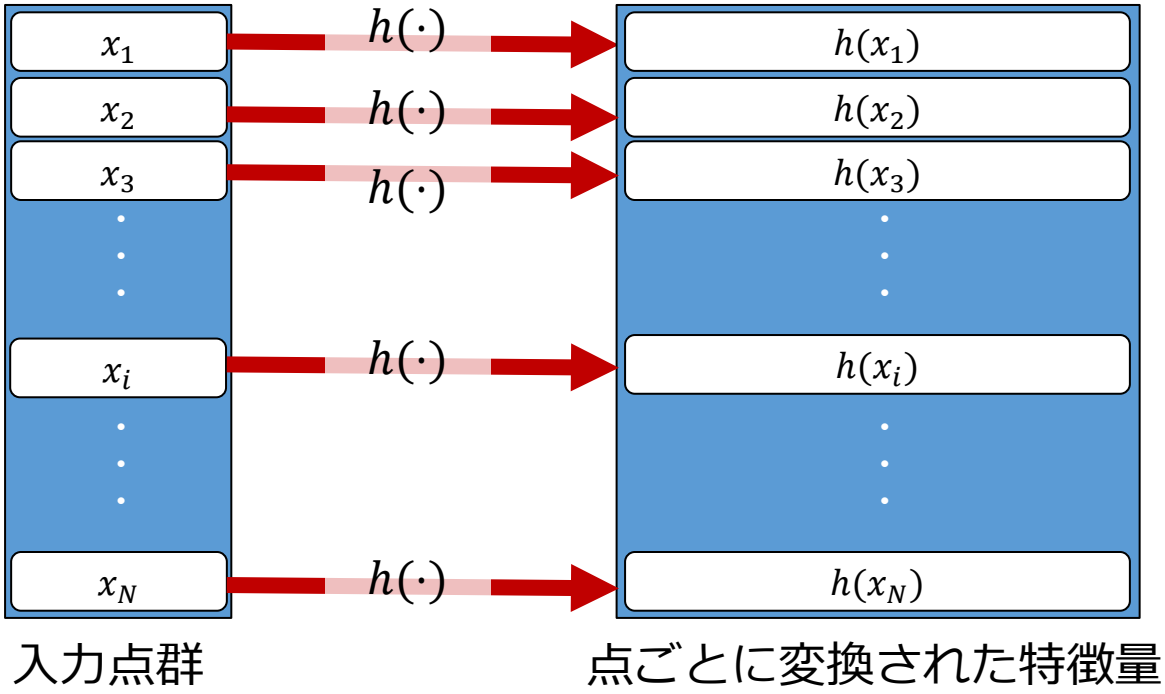
グローバルプーリングであれば順序に依存しない

PointNetの紹介

Symmetric Functionを使うアイデア

ポイント1:

点ごとの変換であれば順序に依存しない



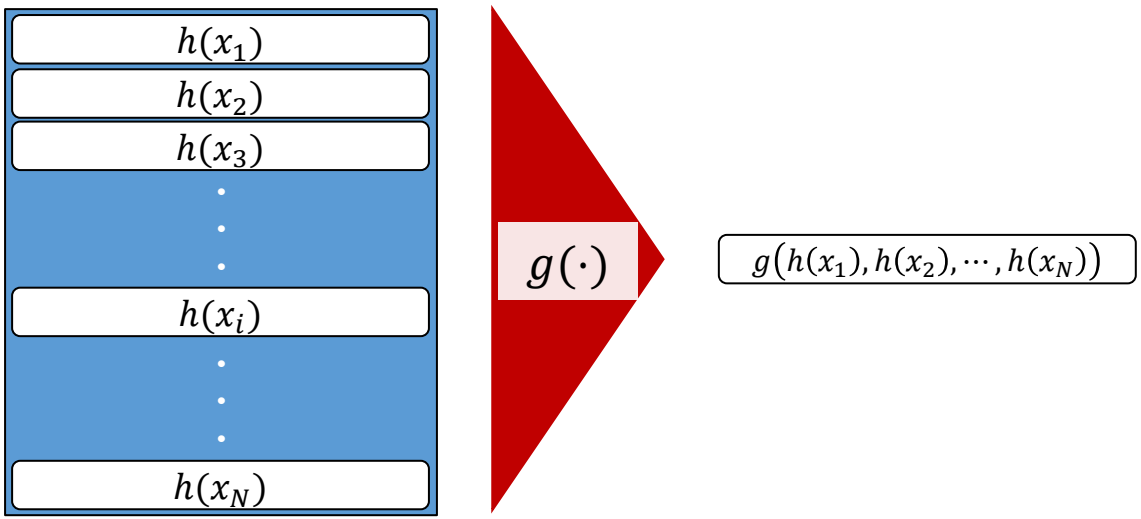
順序に依存しない変換になっている

PointNetの紹介

Symmetric Functionを使うアイデア

ポイント2:

グローバルプーリングであれば順序に依存しない



点ごとに変換された特徴量　　点群全体から集約した特徴量

$g(\cdot)$: 特徴量ベクトルの要素（チャンネル）ごとに，最大値・平均値などを計算

PointNetの紹介

Symmetric Functionを使うアイデア

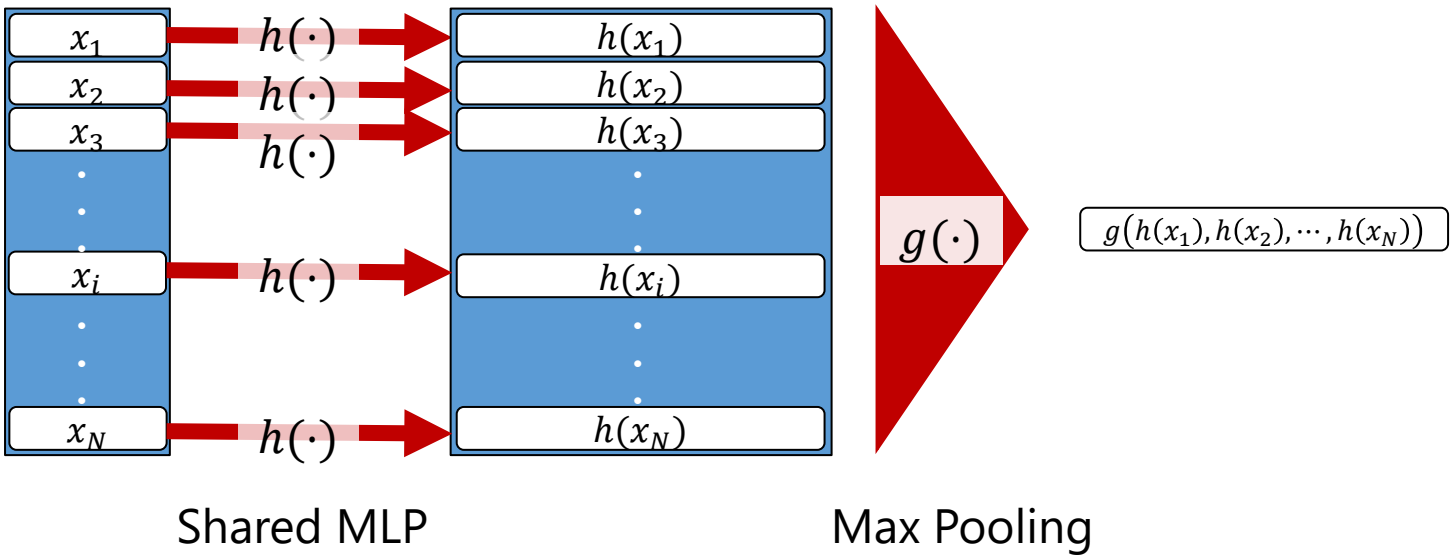
ポイント1: **点ごとの変換**

→ PointNetではShared MLP

（1x1 Convolution, Pointwise Convolutionと同義）

ポイント2: **グローバルプーリング**

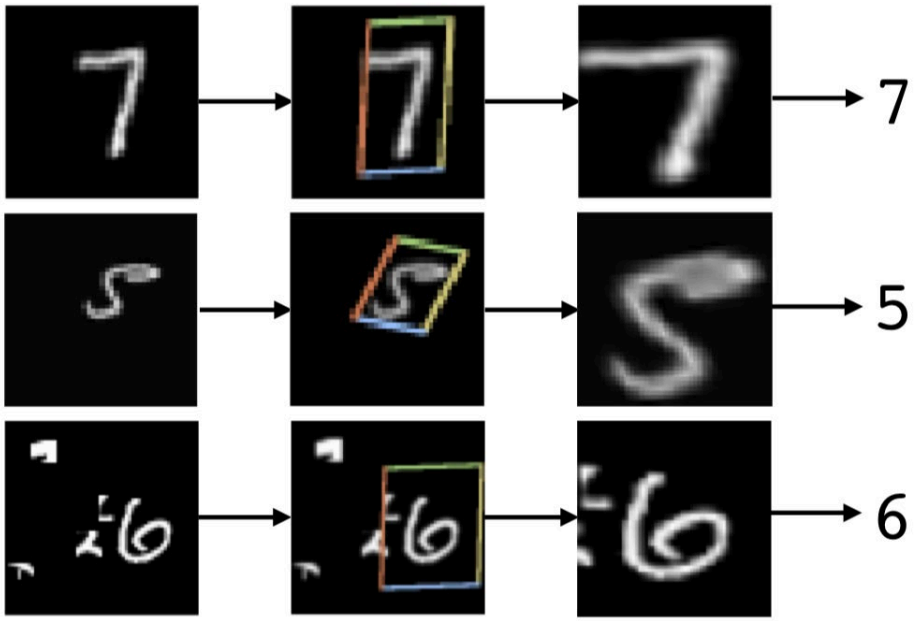
→ PointNetではMax Pooling

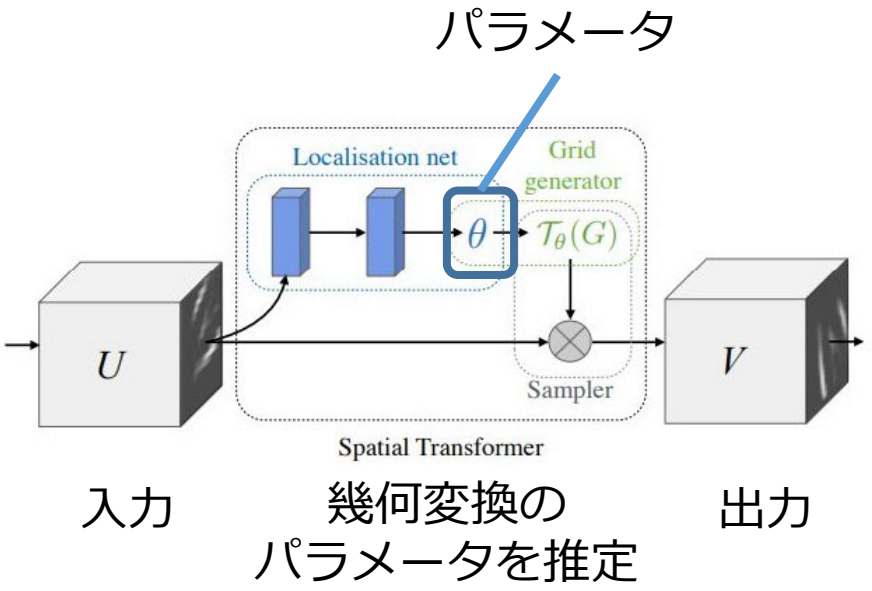


PointNetの紹介

Spatial Transformer Networks (STN)を導入

Spatial Transformer Networks [Jaderberg+, NIPS2015]
幾何変換のパラメータを推定し正規化する構造を持つ





PointNetの紹介

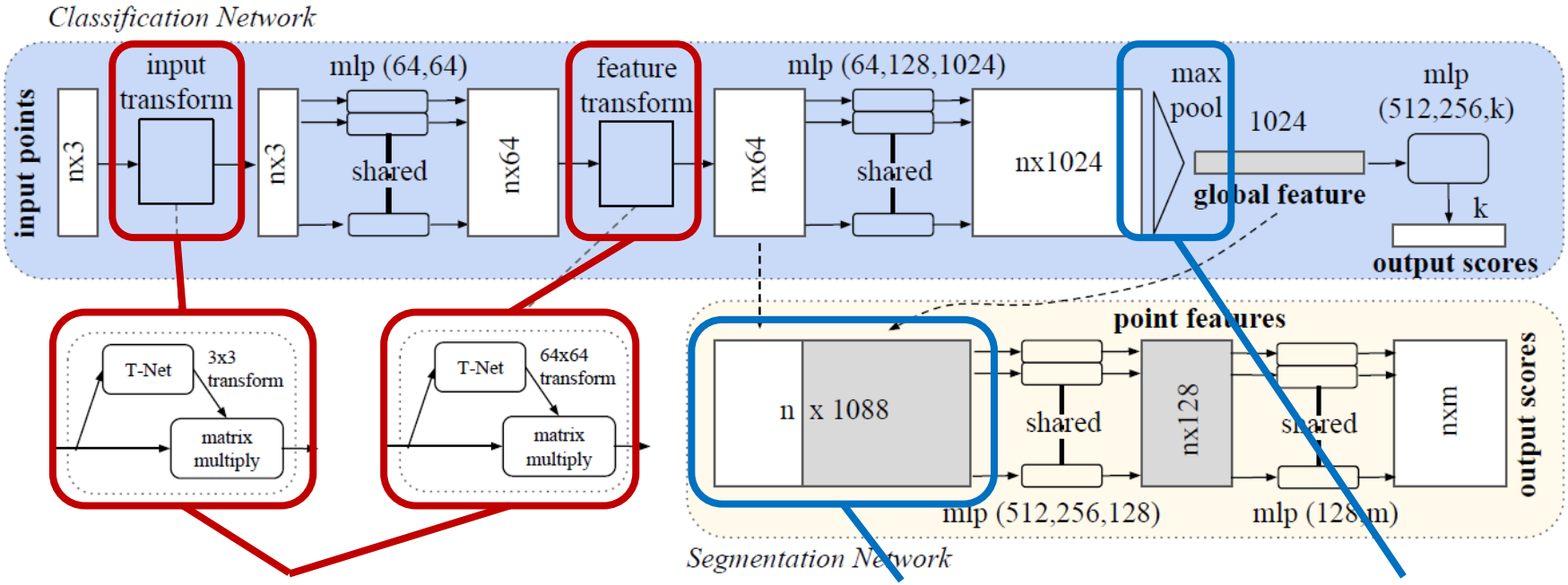
Spatial Transformer Networks (STN)を導入

Spatial Transformer Networks [Jaderberg+, NIPS2015]
幾何変換のパラメータを推定し正規化する構造を持つ

PointNetでの利用法

- Symmetric Functionによって点の順序に不変な
  グローバル特徴量を入力し剛体変換のパラメータを推定
- （剛体変換ではないが）特徴空間でもSTNを適用,
  特徴量空間での変換を行う

PointNetの紹介



T-Net = STN

点ごとの特徴量と
グローバルな特徴量を結合

Max Pooling

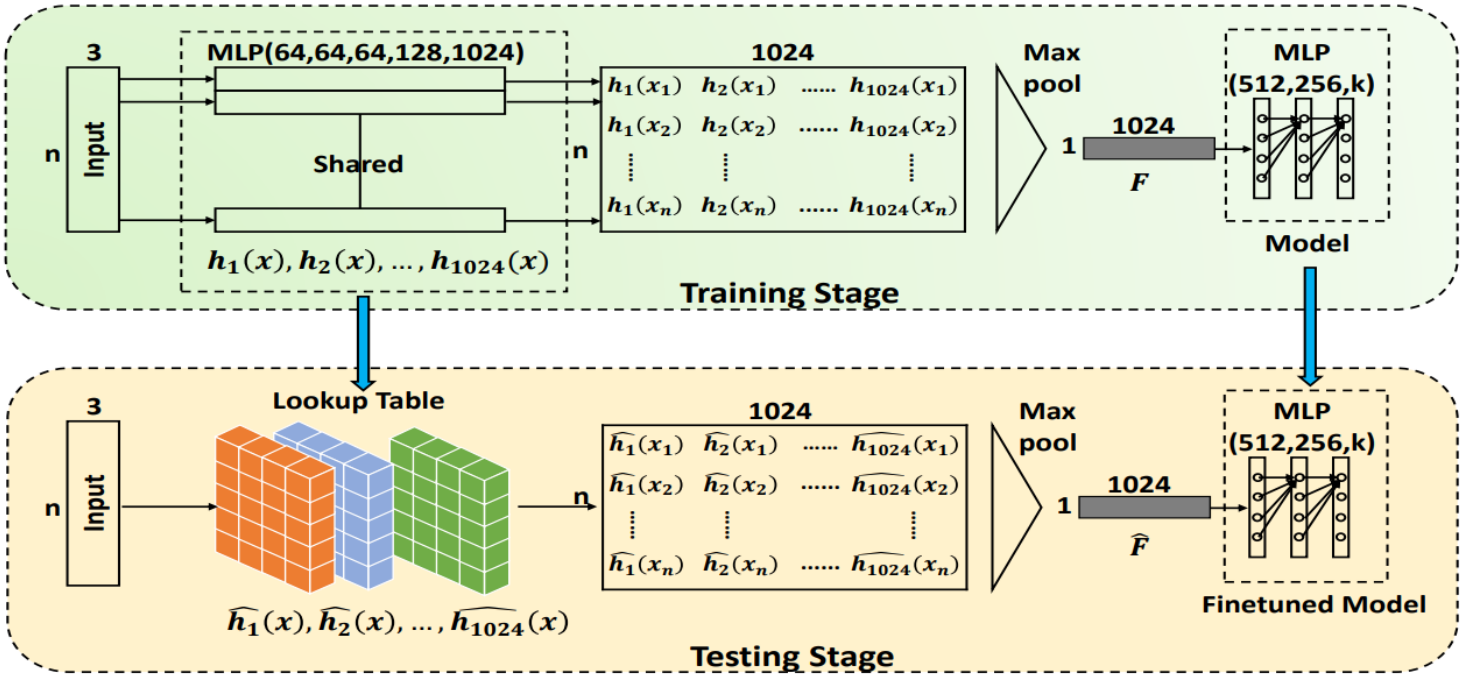
- グローバルな特徴量を使ってクラス分類を実現
- 点ごとの特徴量とグローバルな特徴量を結合してから
  Shared MLPすることで，セマンティックセグメンテーションも実現

高速化手法の紹介

Justlookup [H. Lin+, ICME2019]

PointNetを学習させてからShared MLPをLUTで置き換え、後段のMLPをFine Tuning

推論時にPointNetがGPUで25.3msであるのに対してCPUで1.5msを達成



アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - **点群の畳み込み・Backbone**
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

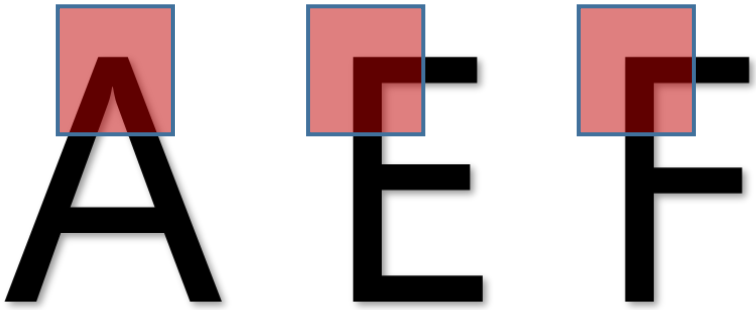
三次元点群に対する畳み込み

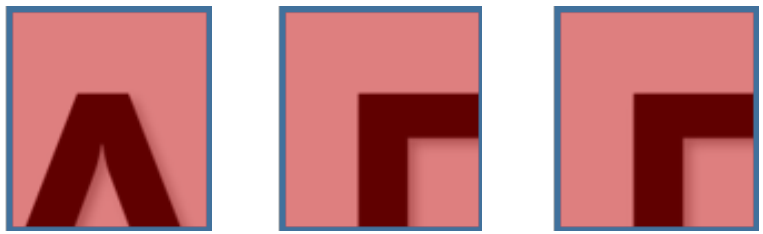
疎な点群に対する畳み込み手法

隣接関係の自明な定義がない
→ PointNet以降，多数の手法が提案された

畳み込みでやりたいこと：**局所特徴量**の抽出

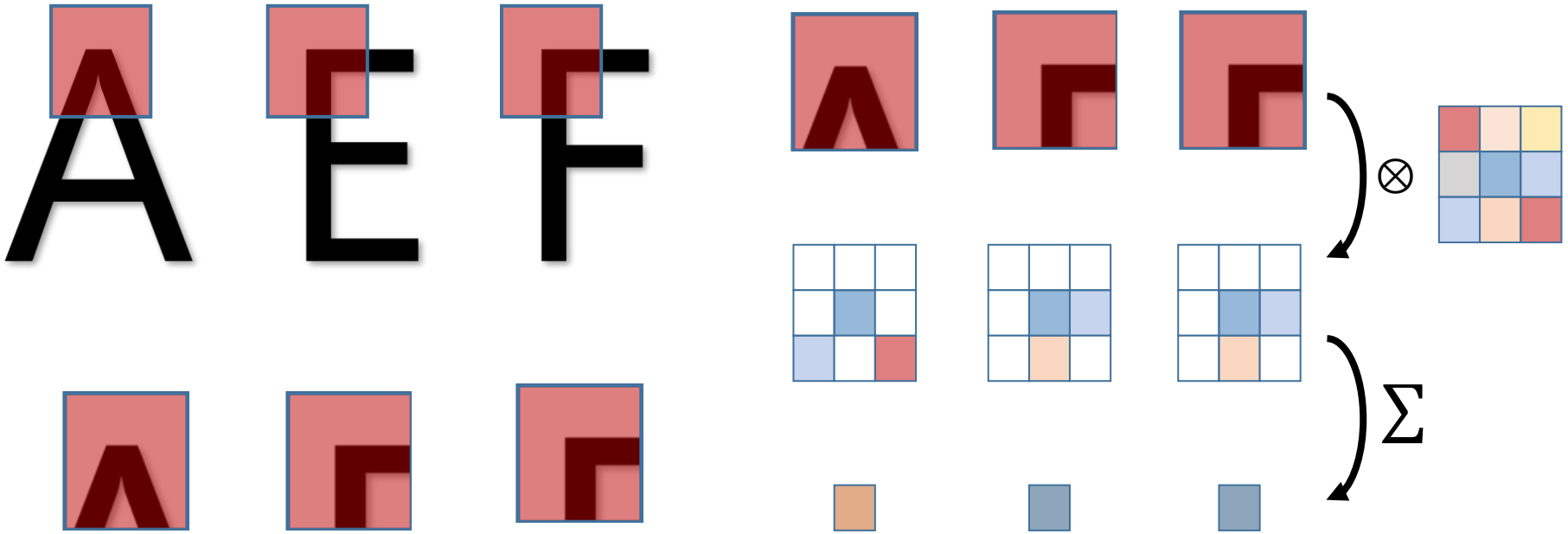
- 畳み込みを用いることで，並進（など）に
  不変な局所特徴量を得ることができる
  （画像における2D畳み込みであれば2D並進不変）
- 例）こちらのカドとあちらのカドで同じ特徴量が抽出





二次元での畳み込み

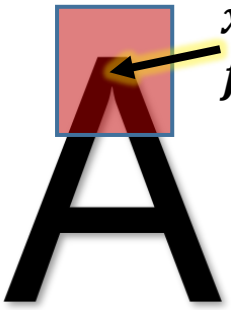
カーネルを重畳，それぞれ掛けて足し合わせる



$$\sum_{\Delta \boldsymbol{x} \in A} \boldsymbol{f}(\boldsymbol{x} + \Delta \boldsymbol{x})^{\top} \boldsymbol{k}(\Delta \boldsymbol{x})$$

二次元での畳み込み

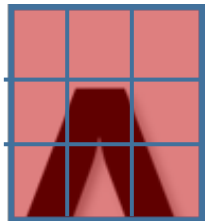
カーネルを重畳，それぞれ掛けて足し合わせる



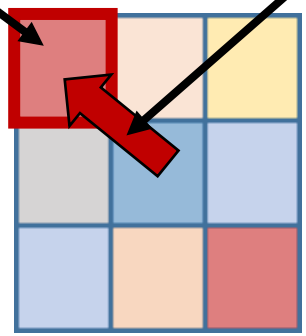
$\boldsymbol{x}$: 着目点
$\boldsymbol{f}(\boldsymbol{x})$: 着目点の入力特徴量

$\boldsymbol{k}(\Delta\boldsymbol{x})$: $\Delta\boldsymbol{x}$での
カーネルの値

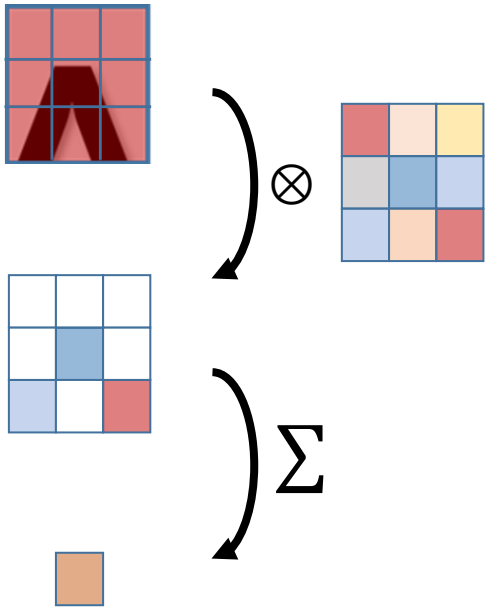
$\Delta\boldsymbol{x}$: カーネル内での
相対座標



A: 着目点周辺の領域



カーネル



$$\sum_{\Delta\boldsymbol{x}\in A} \boldsymbol{f}(\boldsymbol{x}+\Delta\boldsymbol{x})^{\top}\boldsymbol{k}(\Delta\boldsymbol{x})$$

点群における畳み込み

基本方針は二次元での畳み込みを拡張

- $\boldsymbol{f}(\boldsymbol{x})$: 入力特徴量，　$\boldsymbol{f}(\boldsymbol{x}+\Delta\boldsymbol{x})$でそれぞれ定義されている
- $\boldsymbol{k}(\Delta\boldsymbol{x})$: カーネル（一般には$\Delta\boldsymbol{x}$による関数）
- $\Delta\boldsymbol{x} \in A(\boldsymbol{x})$: 近傍点と近傍領域（一般には$\boldsymbol{x}$による集合）
- Agg: 集約関数

$$\mathrm{Agg}_{\Delta\boldsymbol{x}\in A(\boldsymbol{x})}\left(\boldsymbol{f}(\boldsymbol{x}+\Delta\boldsymbol{x})^{\mathsf{T}}\boldsymbol{k}(\Delta\boldsymbol{x})\right)$$

入力特徴量　　カーネル

各点で掛け合わせる

局所領域について集約

点群における畳み込み

基本方針は二次元での畳み込みを拡張

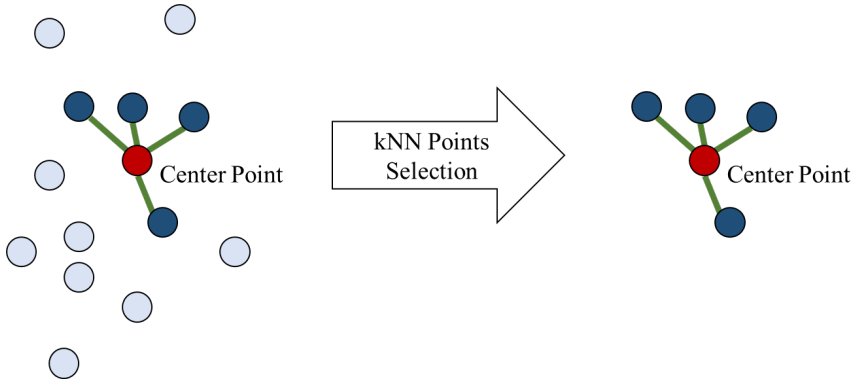
- $\boldsymbol{f}(\boldsymbol{x})$: 入力特徴量，$\boldsymbol{f}(\boldsymbol{x}+\Delta\boldsymbol{x})$でそれぞれ定義されている
- $\boldsymbol{k}(\Delta\boldsymbol{x})$: カーネル（一般には$\Delta\boldsymbol{x}$による関数）
- $\Delta\boldsymbol{x} \in A(\boldsymbol{x})$: 近傍点と近傍領域（一般には$\boldsymbol{x}$による集合）
- Agg: 集約関数

手法ごとに違うのは

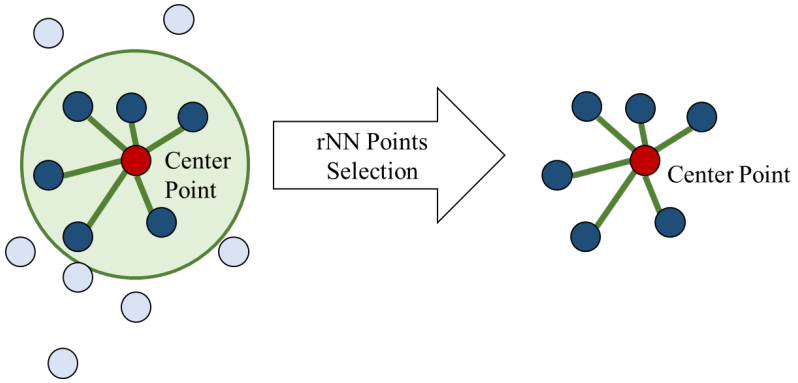
- **近傍をどう決めるか**
- **カーネルをどう決めるか**（≒どう学習可能にするか）
- **どう集約するか**

近傍をどう決めるか

多くは**kNN** (k-Nearest Neighbor) か
rNN (radius-Nearest Neighbor)
　メッシュが入力の場合は接続関係を利用する場合も



kNN: 近傍点k個を選択



rNN: 半径r以内の点を選択

近傍をどう決めるか

各手法の特徴

- kNN: シンプル・近傍が同じ点数なので**扱いやすい**
- rNN: **形状に伴った近傍領域の設定ができる**が
  各点の近傍の点数がバラバラになるので**扱いにくい**

典型的な計算方法

- kNN: 距離行列を計算しTop-kを選択
- rNN: 距離行列を計算しスレッショルド

  真にrNNで近傍を設定すると、最悪ケースで全点が半径r内に入ってしまい計算コスト大

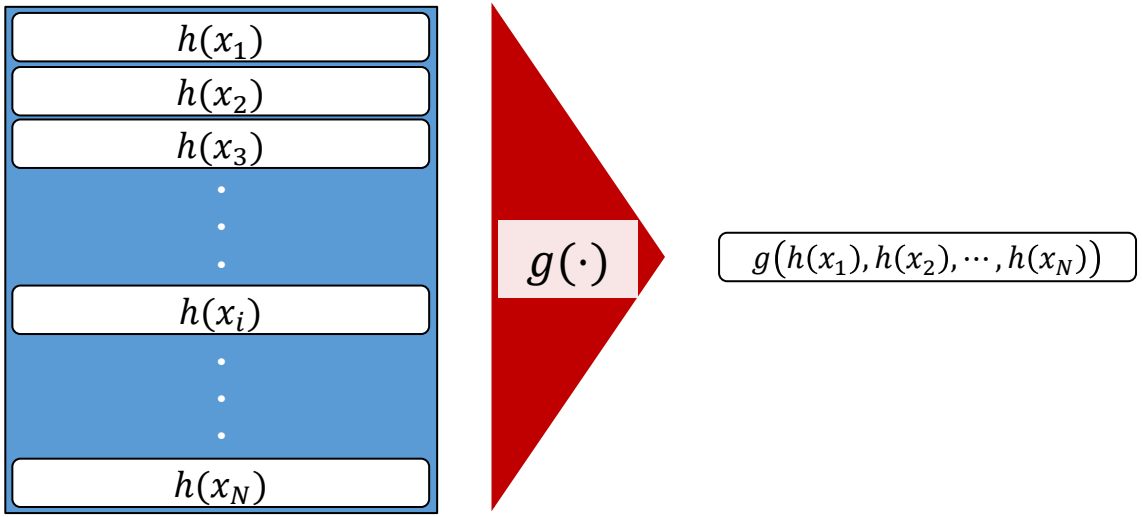
  → 予めkNNで近傍を切ってからrNNとする例が多い

集約をどうするか

順不同な関数であればOK

PointNetでMax-poolingを使うのと同じ

- Sum: **元の畳み込みに近い構造**になる
- Max, Mean: **個数が変化しても対応**できる



局所点群の点ごとの特徴量

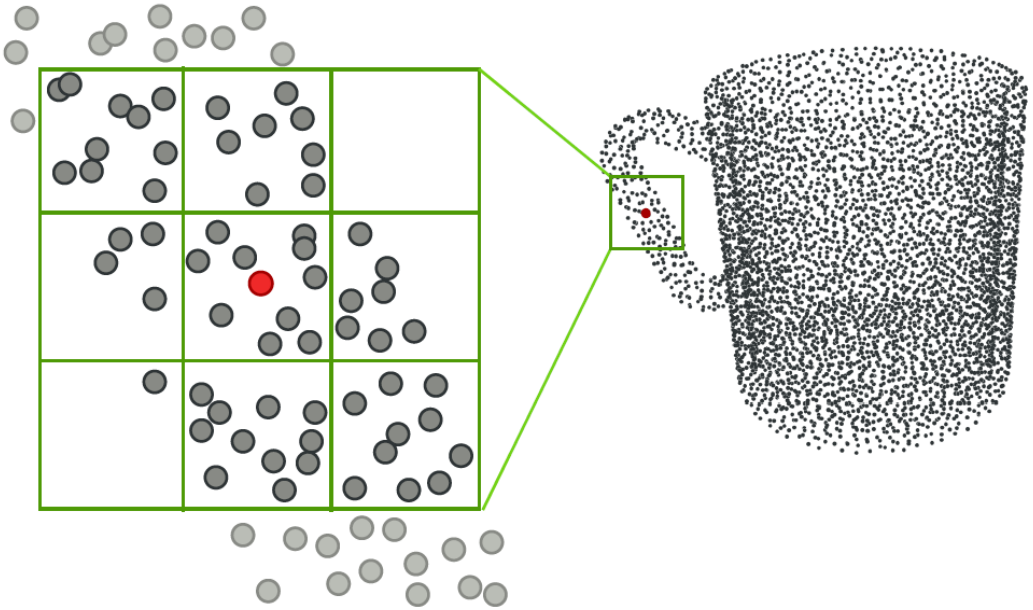
集約した特徴量

畳み込みの手法の紹介

Pointwise Convolution [B. S. Hua+, CVPR2018]

1. 点ごとにボクセルグリッドでカーネルを定義
2. ボクセルごとに特徴量の平均を求め，畳み込み

演算は高速だが剛体変換によって変化

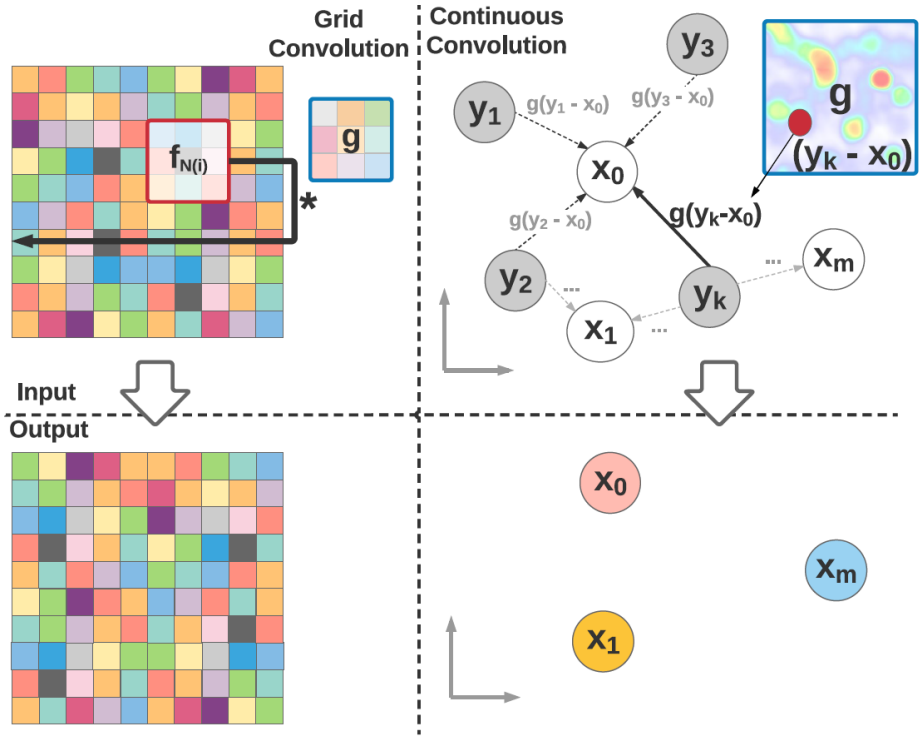


畳み込みの手法の紹介

Parametric Continuous Convolutions [S. Wang+, CVPR2018]

非グリッドな環境でのカーネルを
MLPで学習

1. kNNやrNNで
   局所領域を選択
2. 注目点との相対
   座標→その点での
   重みを出力



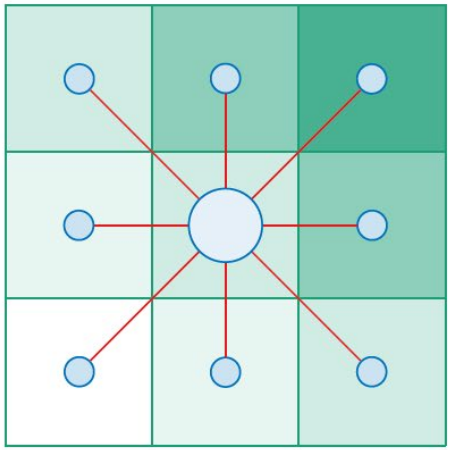
Grid Convolution

Continuous
Convolution

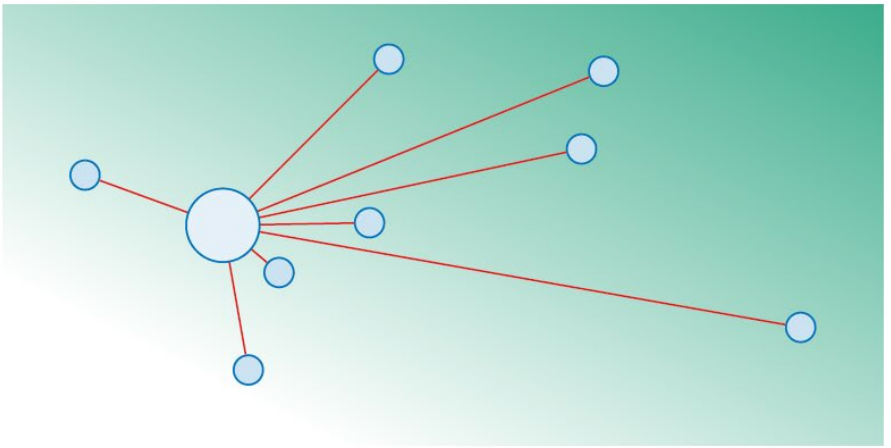
畳み込みの手法の紹介

Flex-convolution [F. Groh+, ACCV2018]

1. 畳み込む点周りで近傍点を選択
2. 相対位置をアフィン変換してカーネルの各要素に割り当てる
3. カーネルで畳み込み



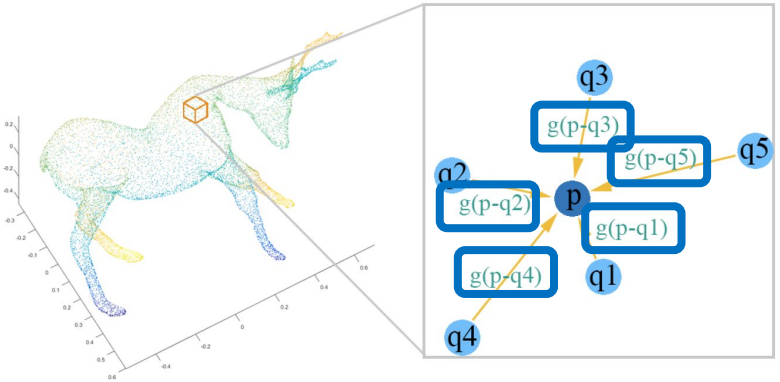
二次元画像での畳み込み



三次元点群でのFlex-convolution
中心点との相対位置によって重みを変える

畳み込みの手法の紹介

SpiderCNN [Y. Xu+, ECCV2019]
相対関係を入力，
重みを出力する
関数$g_w(x,y,z)$を学習



$$g_w(x,y,z) = g_{w^S}^{Step}(x,y,z) \cdot g_{w^T}^{Taylor}(x,y,z),$$

$$g_{w^S}^{Step}(x,y,z) = w_i^S \text{ if } r_i \leq \sqrt{x^2+y^2+z^2} < r_{i+1},$$

一定半径の球の内部にあるかの指示関数

$$\begin{aligned} g_{w^T}^{Taylor}(x,y,z) = {} & w_0^T + w_1^T x + w_2^T y + w_3^T z + w_4^T xy + w_5^T yz + w_6^T xz + w_7^T x^2 \\ & + w_8^T y^2 + w_9^T z^2 + w_{10}^T xy^2 + w_{11}^T x^2 y + w_{12}^T y^2 z + w_{13}^T yz^2 \\ & + w_{14}^T x^2 z + w_{15}^T xz^2 + w_{16}^T xyz + w_{17}^T x^3 + w_{18}^T y^3 + w_{19}^T z^3. \end{aligned}$$

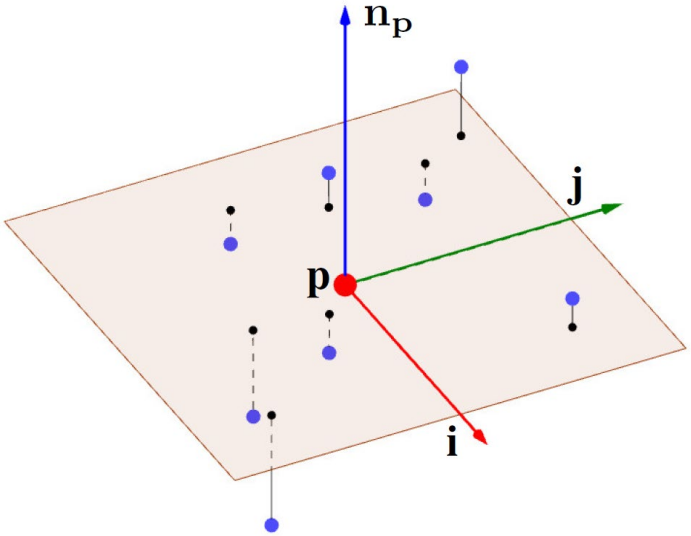
連続関数に対するx,y,zの3次までのテイラー展開

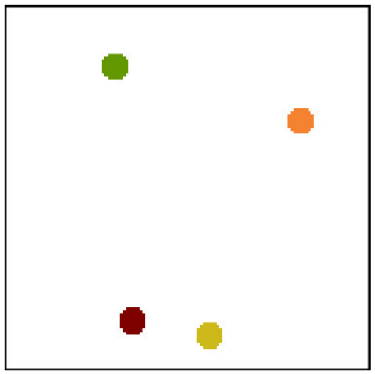
畳み込みの手法の紹介

Tangent Convolution

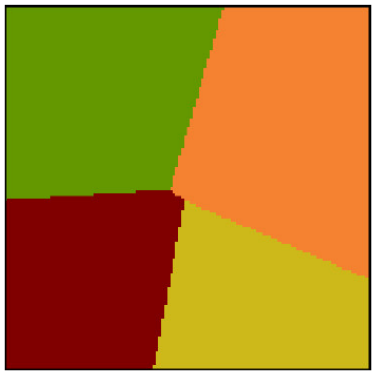
[M. Tatarchenko+, CVPR2018]

1. 各点の接平面を求める
2. 接平面上に近傍点を投影
3. 接平面を埋めるように補間
4. 2D CNNで畳み込み

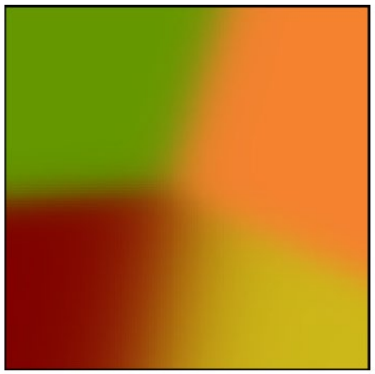




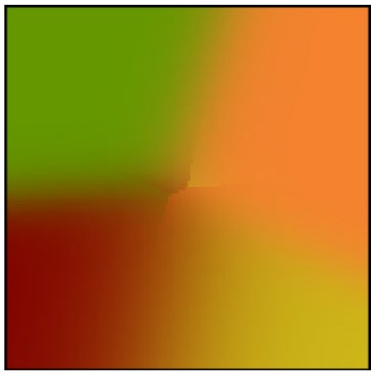
三次元点の投影



Nearest Neighbor



Full Gaussian Mixture



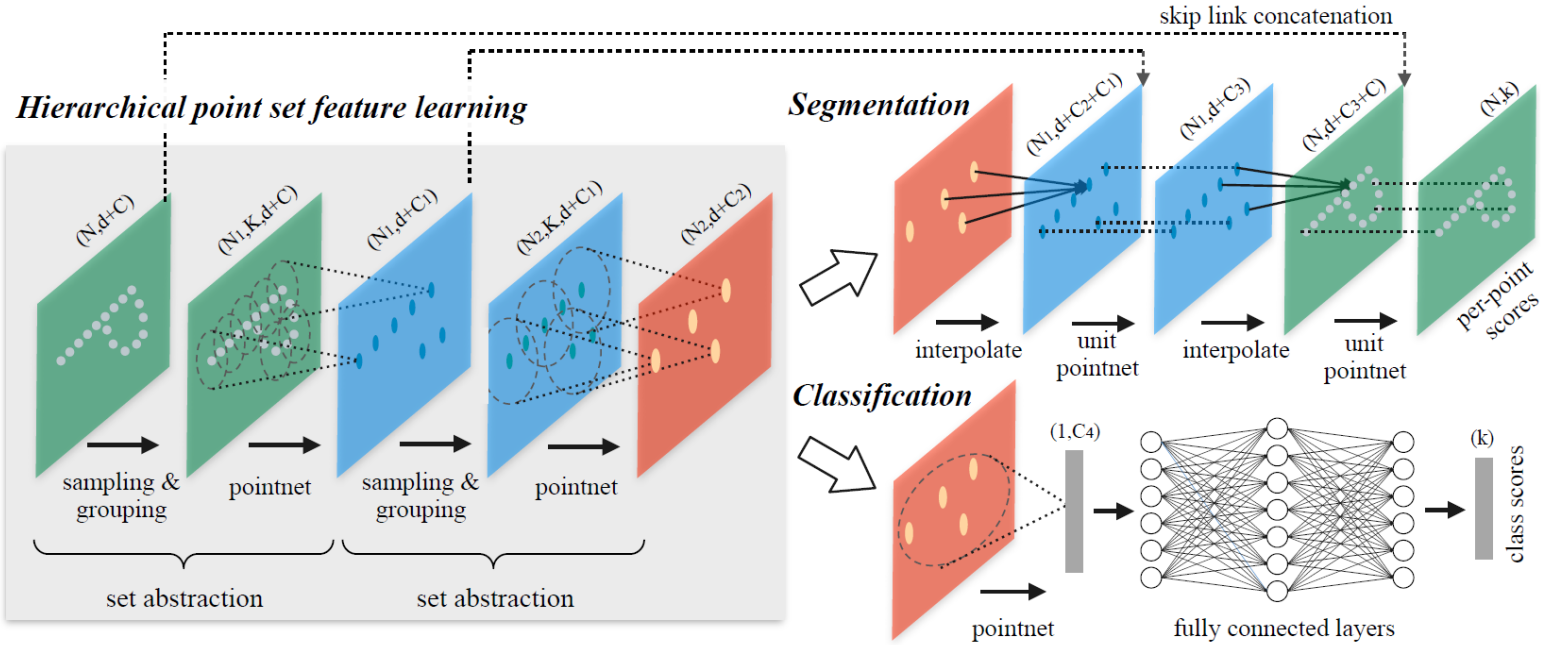
Top-3 Neighbors Gaussian Mixture

畳み込みの手法の紹介

PointNet++ [C. R. Qi+, NIPS2017]

PointNet著者らによる点群畳み込み手法

- Sampling Layer: 重心位置を選択
- Grouping Layer: 近傍点群を抽出
- PointNet Layer: 局所特徴量を計算

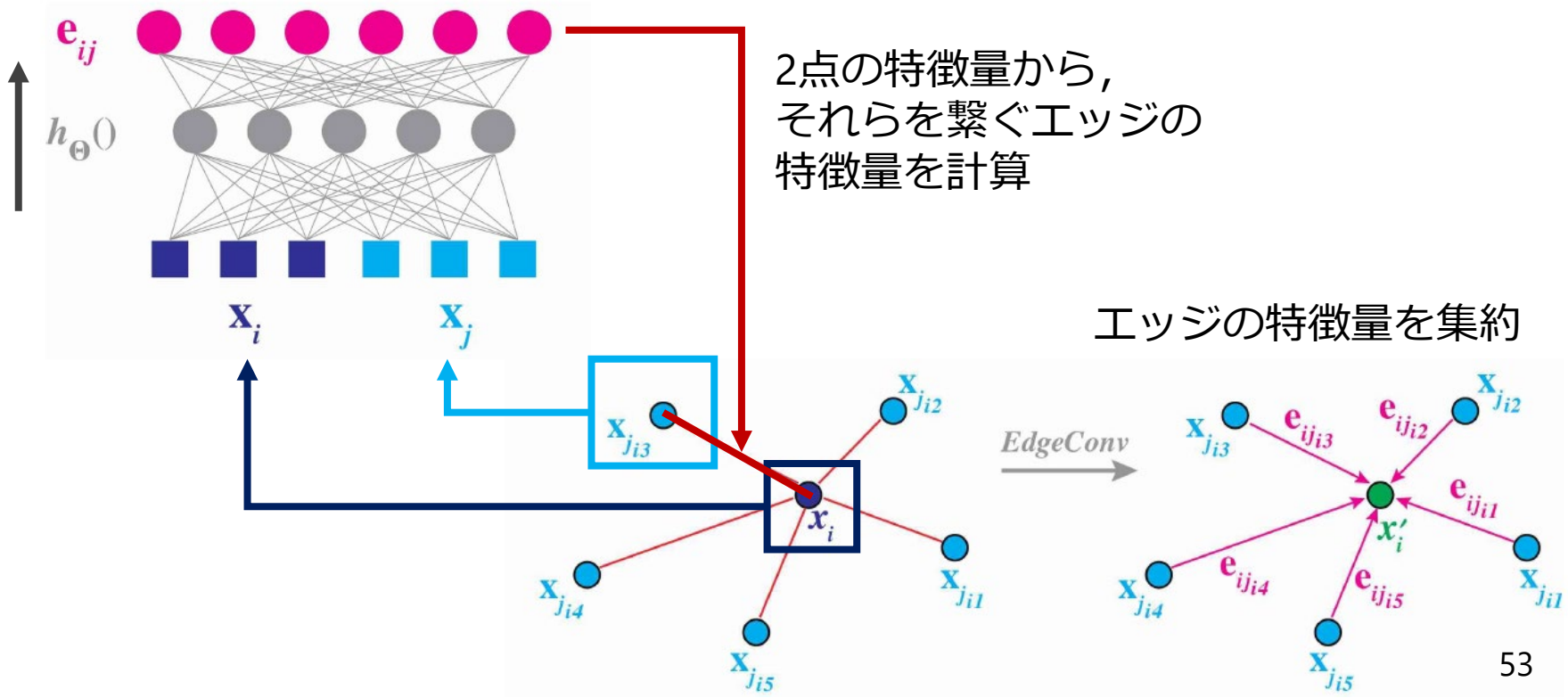


畳み込みの手法の紹介

Dynamic Graph CNN (DGCNN) のEdgeConv

[Y. Wang+, ACM ToG, 2019]

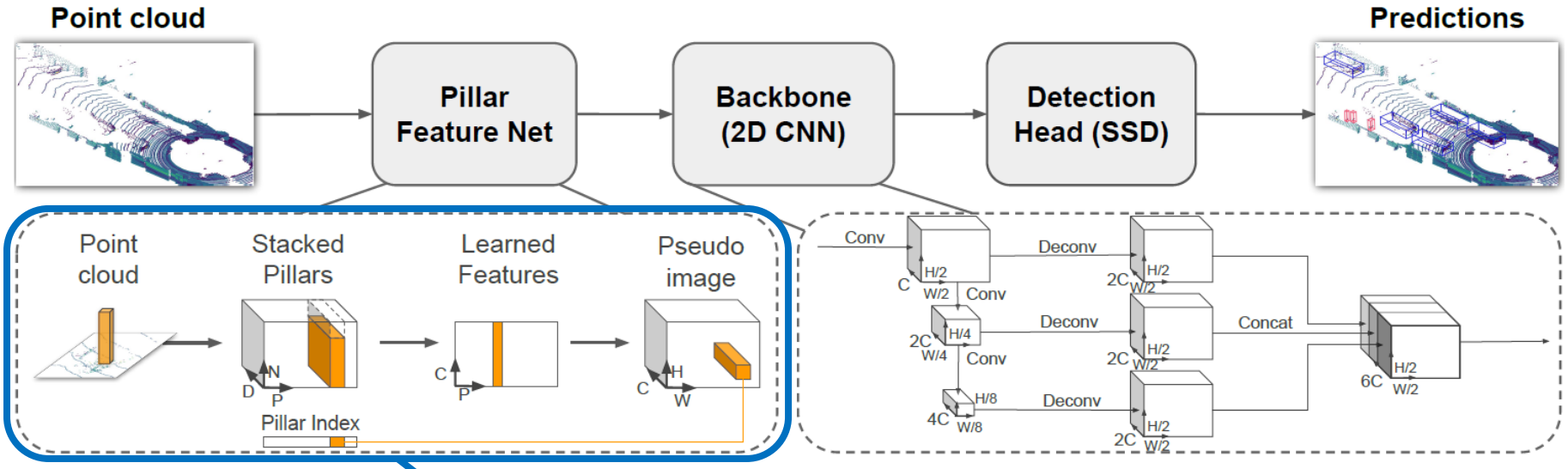
1. エッジごとに点ペアから特徴量をShared MLPで計算
2. ノードごとにエッジの特徴量を集約



畳み込みの手法の紹介

PointPillars [A. H. Lang+, CVPR2019]

- PointPillar Networkで3D→2D Bird's Eye View (BEV)に変換
  （地面方向が既知，という強い仮定が必要）
- 2D CNNで処理し2DでDetection （3D BBを推定）
- 高速で高精度な物体検出を実現

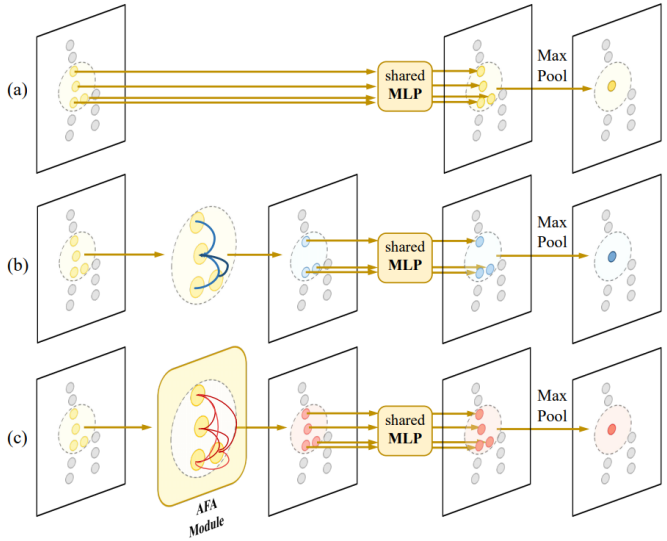


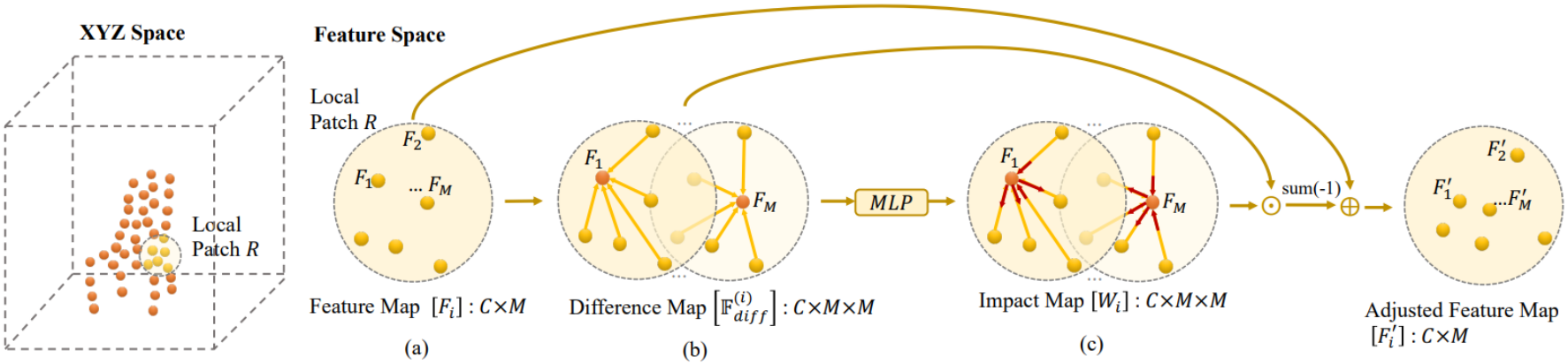
点群から柱状領域を抽出→特徴量を2Dにマップ

畳み込みの手法の紹介

PointWeb [H. Zhao+, CVPR2019]

- 局所領域内の点の相対関係を利用して特徴量を重み付け
- 局所領域の全点ペアについて考慮して特徴量を更新（Adaptive Feature Adjustmentモジュール）

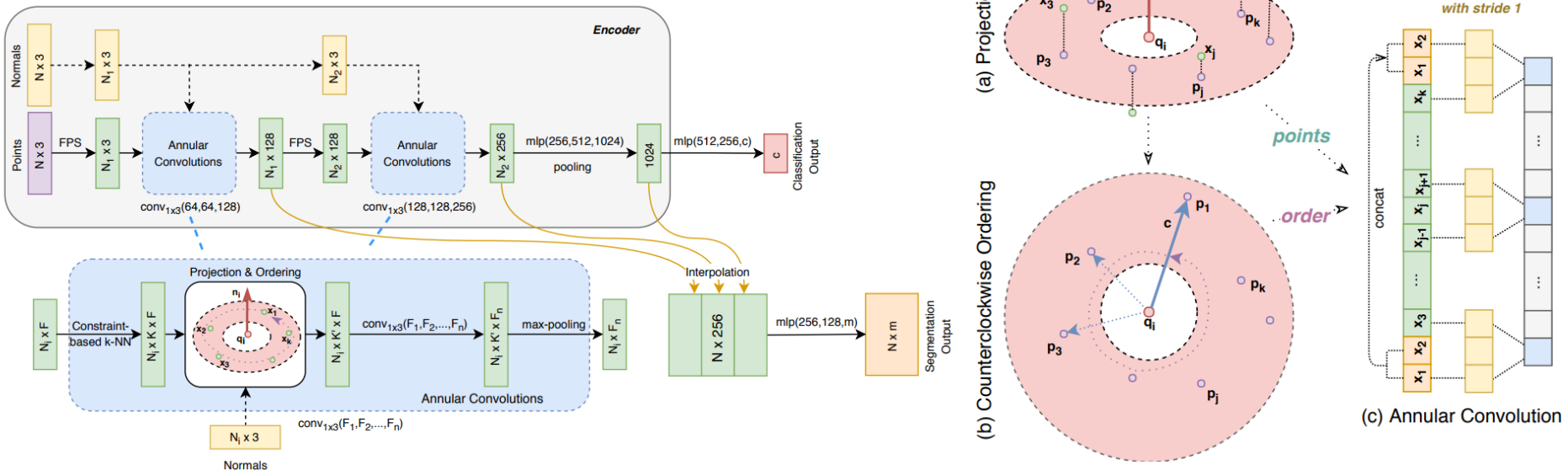




畳み込みの手法の紹介

A-CNN [A. Komarichev+, CVPR2019]

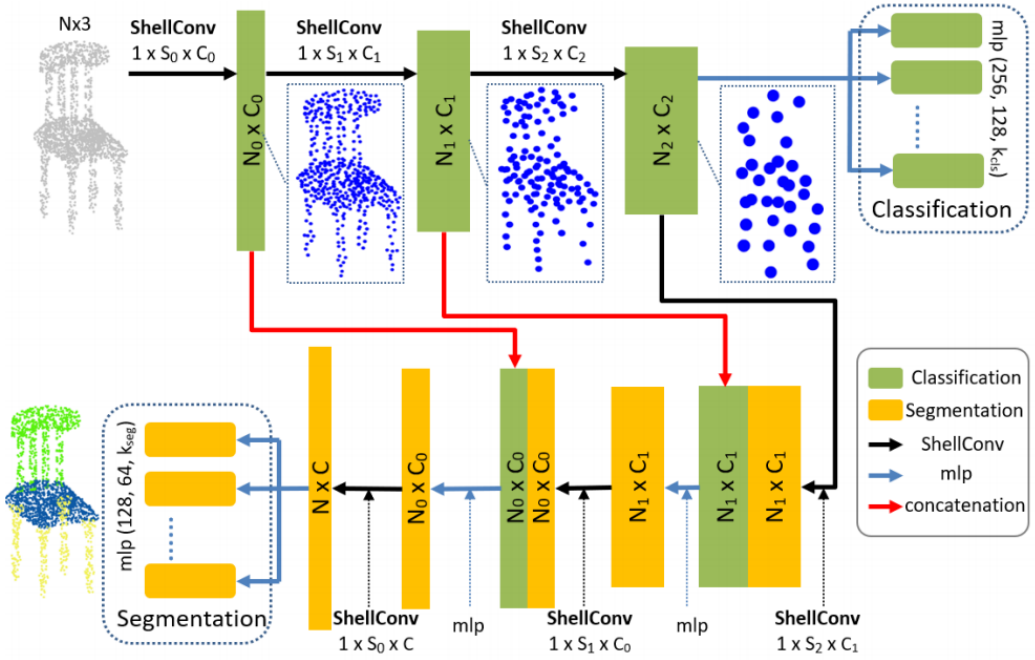
1. 接平面上に投影してから反時計回りに順序付け
2. 1D CNNで畳み込む

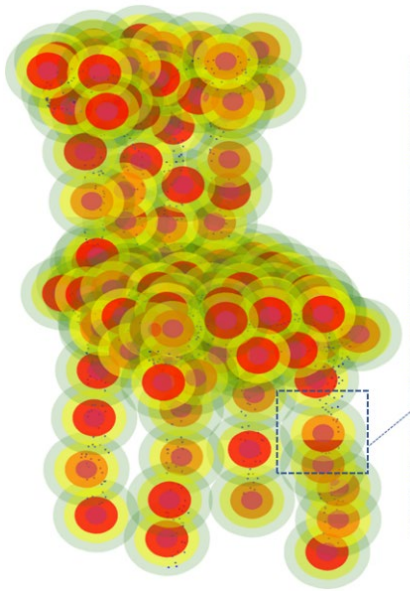


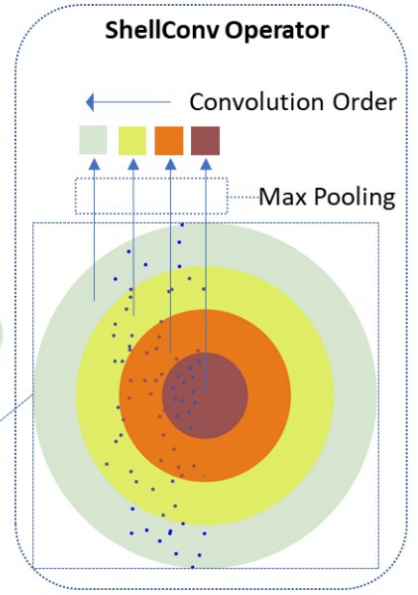
畳み込みの手法の紹介

ShellNet [Z. Zhang+, ICCV2019]

1. 注目点からの距離に応じたビンで区切る
2. ビンごとに特徴量をMax-pooling
3. 結合して1D CNNで畳み込み



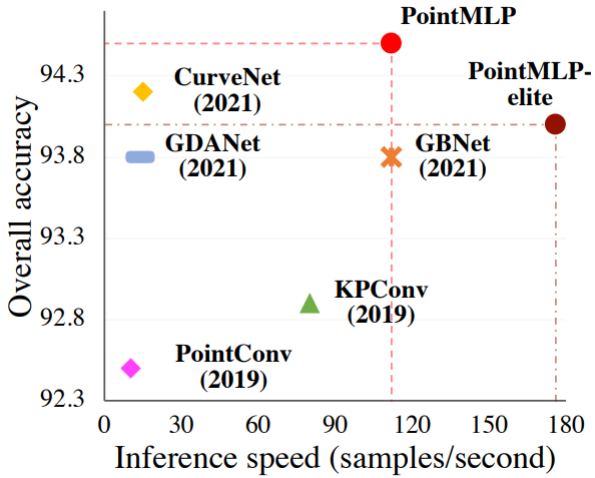


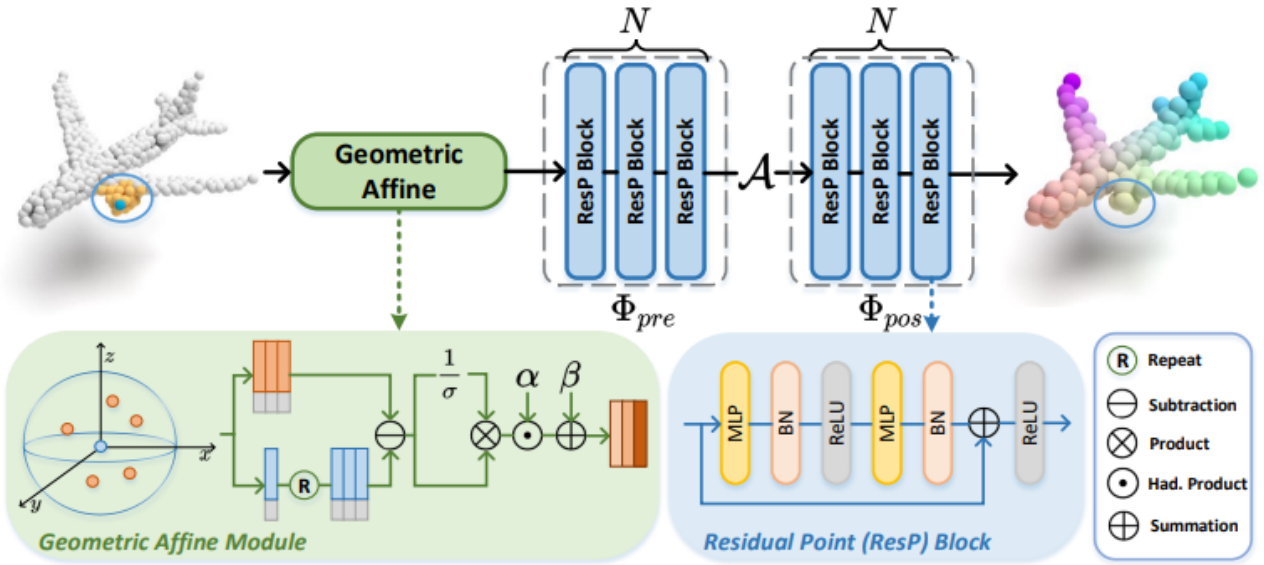


畳み込みの手法の紹介（現時点オススメ）

PointMLP [X. Ma+, ICLR2022]

- 精度・推論/学習速度ともに優れる
- 局所点群をアフィン変換する軽量なモジュールの後にResidual接続を用いた深めのShared-MLPとPoolingで処理





試すには？

- PointNet++/EdgeConv/SpiderConvを試す
- 2Dに変換できる場合は2D CNNも検討
  - BEW
  - Front View
  - (Depth Image, Multi-view, etc...)
- 目的に応じた畳み込みを検討
  - 無理に最新手法を追いかける必要があるかは微妙
  - TransformerやMLP Mixerの研究進展が割とそのまま流用できる問題設定

グラフNNとの関係

グラフ畳み込みネットワーク（Graph Convolutional Networks, GCN）の文脈では，**Message Passing**として整理されることが多い

- 基本的な発想は同じ
- **近傍領域が自明・グラフは固定**（1-hopで接続）
- 近傍点との関係による特徴量がエッジに乗る，それを各点に集約する，という発想

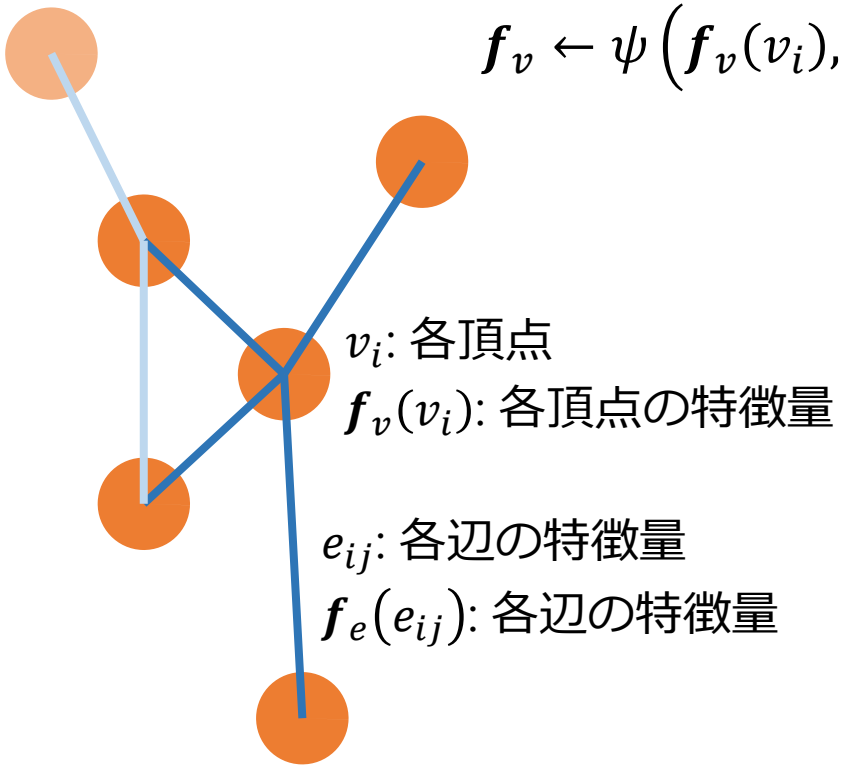
$$\boldsymbol{f}_v \leftarrow \psi\left(\boldsymbol{f}_v(v_i), \mathrm{Agg}_{e_{ij}\in E(v_i)}\left\{\phi\left(\boldsymbol{f}_v(v_i), \boldsymbol{f}_v(v_j), \boldsymbol{f}_e(e_{ij})\right)\right\}\right)$$

- v_i: 各頂点
- $\boldsymbol{f}_v(v_i)$: 各頂点の特徴量
- e_{ij}: 各辺
- $\boldsymbol{f}_e(e_{ij})$: 各辺の特徴量

グラフNNとの関係

グラフ畳み込みネットワーク（Graph Convolutional Networks, GCN）の文脈では，**Message Passing**として整理されることが多い

$$\boldsymbol{f}_v \leftarrow \psi\left(\boldsymbol{f}_v(v_i), \mathrm{Agg}_{e_{ij}\in E(v_i)}\left\{\phi\left(\boldsymbol{f}_v(v_i), \boldsymbol{f}_v(v_j), \boldsymbol{f}_e(e_{ij})\right)\right\}\right)$$

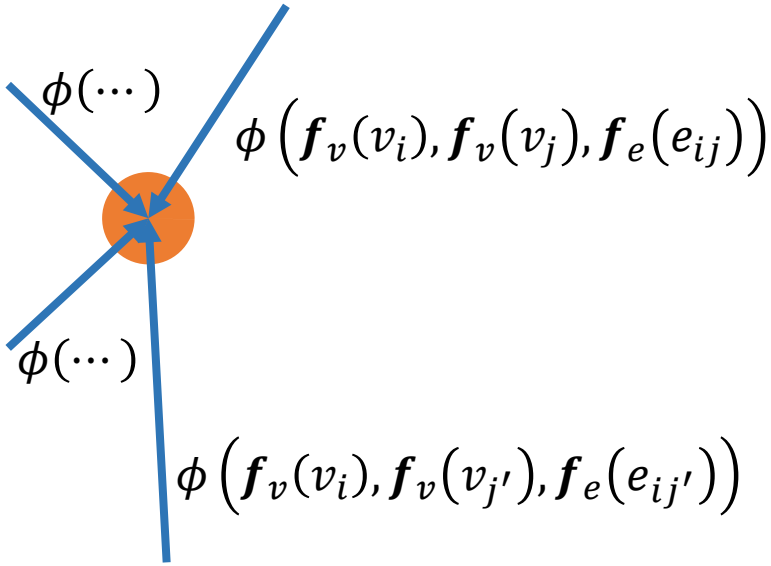


v_i: 各頂点

$\boldsymbol{f}_v(v_i)$: 各頂点の特徴量

e_{ij}: 各辺の特徴量

$\boldsymbol{f}_e(e_{ij})$: 各辺の特徴量



Transformerとの関係

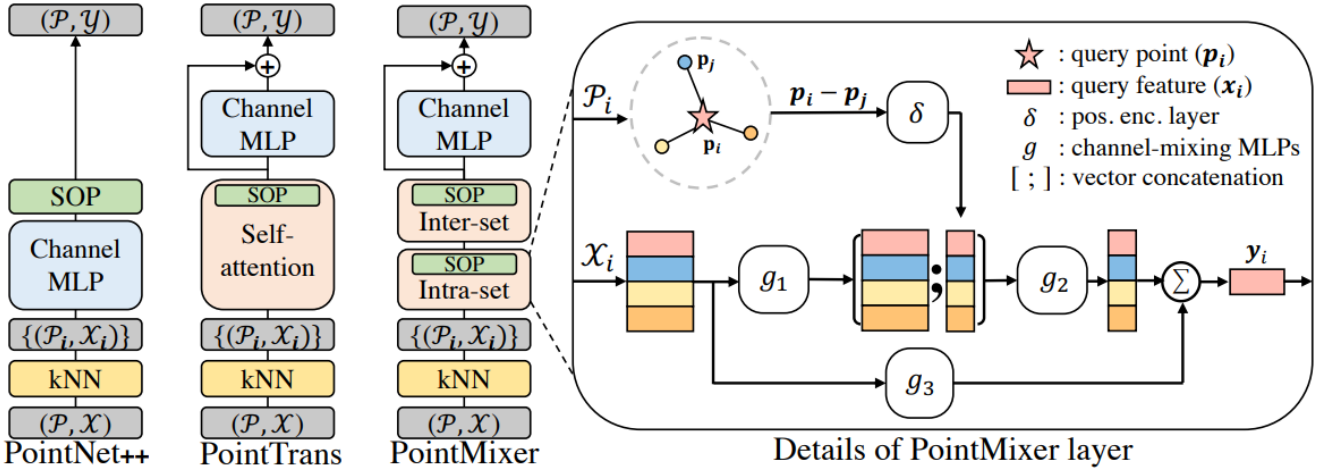
入力が集合（＝順不同）なので，基本的に
点群との相性は良い

- ただし入力点数が多いため，サブサンプリングなどで点数削減が必要
- ある意味ではTransformerにPositional Encodingだけを入力しているようなもの

- Point Transformer [H. Zhao+, ICCV2021]や Pointformer [X. Pan+, CVPR2021]など，利用例が現れつつある

MLP Mixerとの関係

- こちらもおそらく相性は良い
  順不同にするか順序を入れる工夫が必要
- PointMixer [J. Choe+, arXiv, 2021]などの利用例
  - この論文中の図がわかりやすい.
  - 広く解釈するとTransformerもMLP Mixerも点群畳み込みに近い枠組みで整理できる.
    このあたりは集合データ処理として今後整理が進むと期待



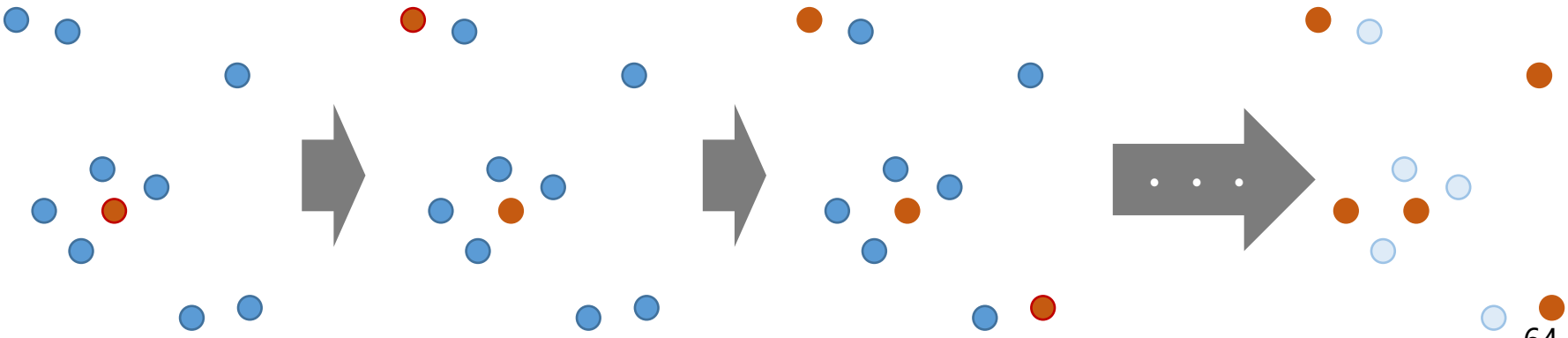
サブサンプリング手法

サブサンプリングの必要性

- 2D CNN同様に，点群も階層的に処理したい
- 点群にグリッド構造はないためサブサンプリングが必要

よく使われる手法: **Furthest Point Sampling (FPS)**

- PointNet/PointNet++で採用，それ以降の手法でも広く利用されている
- 選ばれている点群から最も遠い点を逐次選択していく

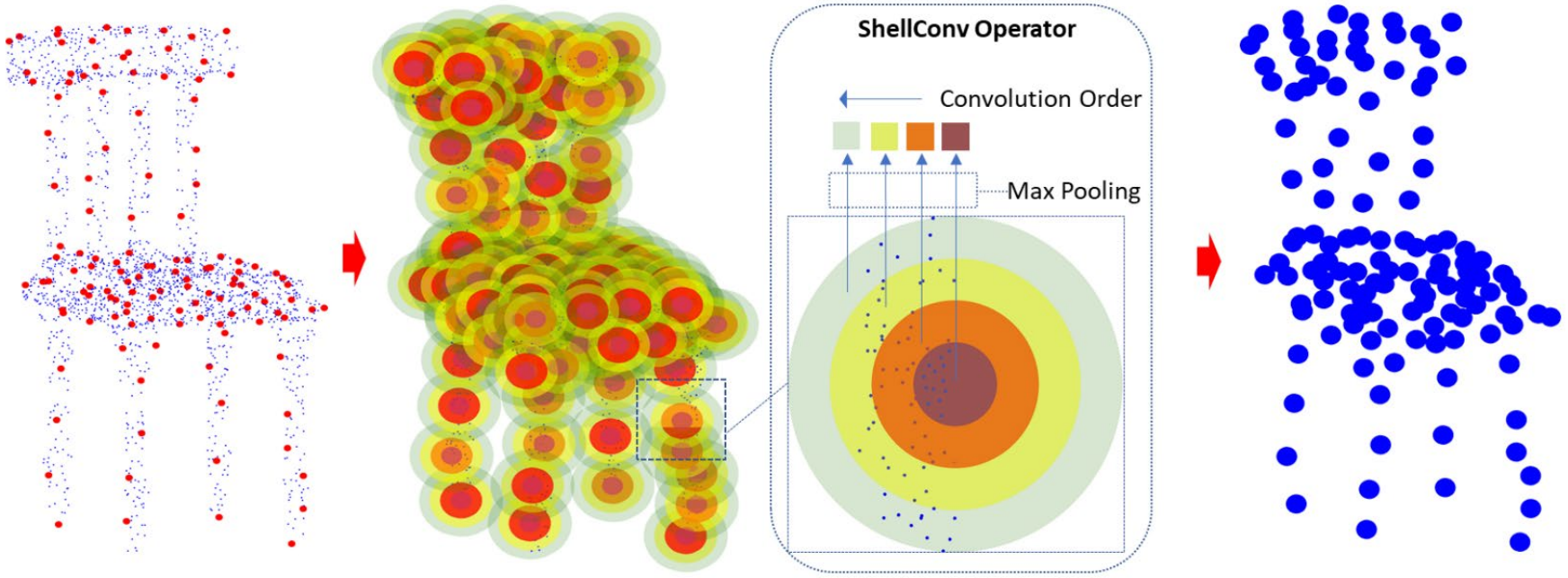


サブサンプリング手法の紹介

ShellNet [Z. Zhang+, ICCV2019]

サンプリング手法: ランダムに選択

ShellConvによる実験で，サンプリングがランダムでも精度が落ちないことを確かめた



サブサンプリング手法の紹介

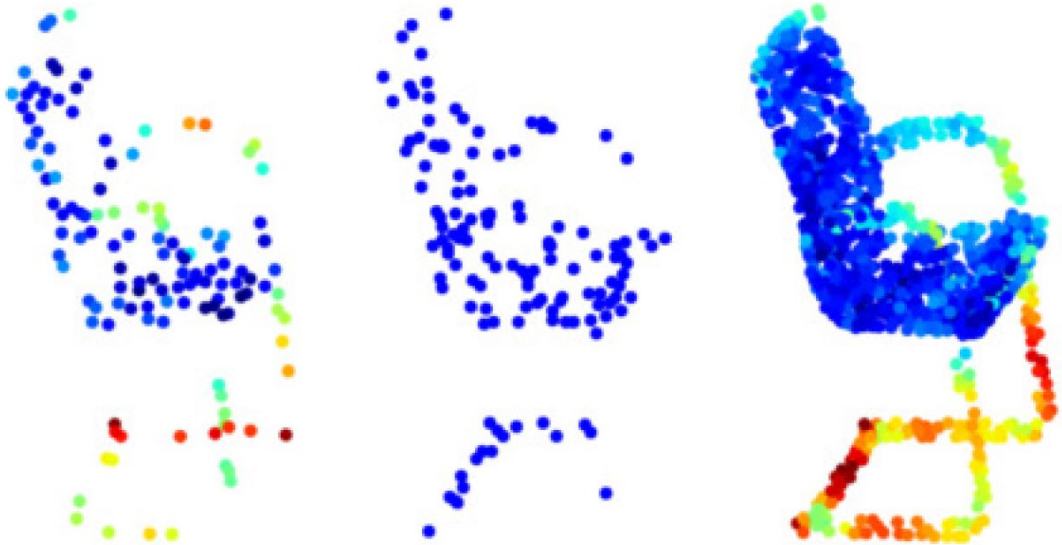
Flex-convolution [F. Groh+, ACCV2018]

サンプリング手法:

Inverse Density Importance Sub-Sampling (IDISS)

IDISSは点数に比例する計算量

（FPSは点数の二乗に比例する計算量なので効率が良い）



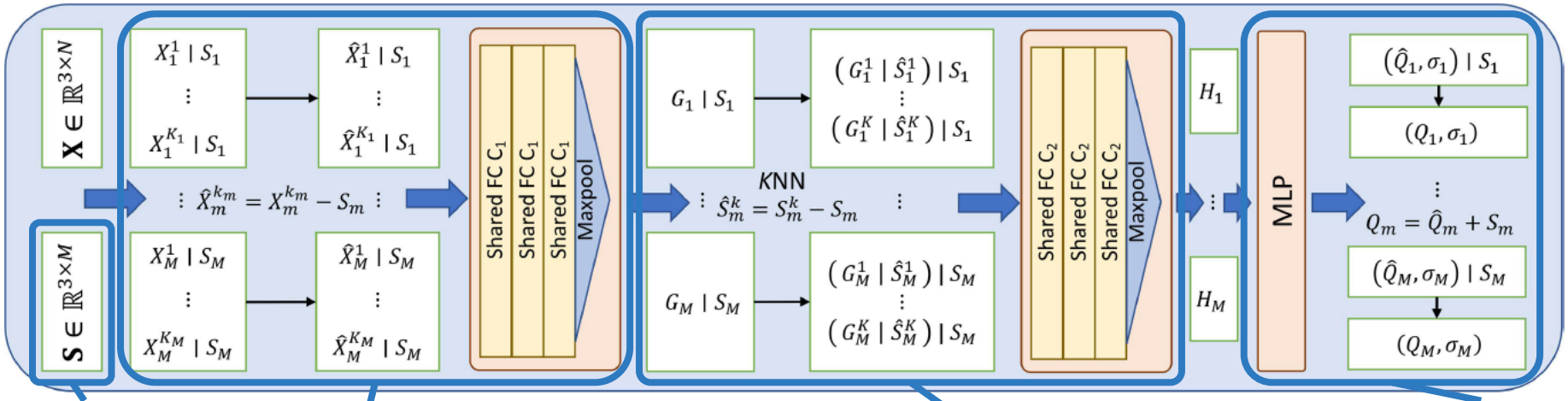
IDISS　　ランダム　　元の点群

サブサンプリング手法の紹介

USIP [J. Li+, ICCV2019]

サンプリング手法: FPS + 補正
（位置合わせのための特徴点検出手法）

1. FPSでサンプリング，その近傍で局所領域を設定
2. 局所領域内・局所領域同士の関係から代表点を再設定



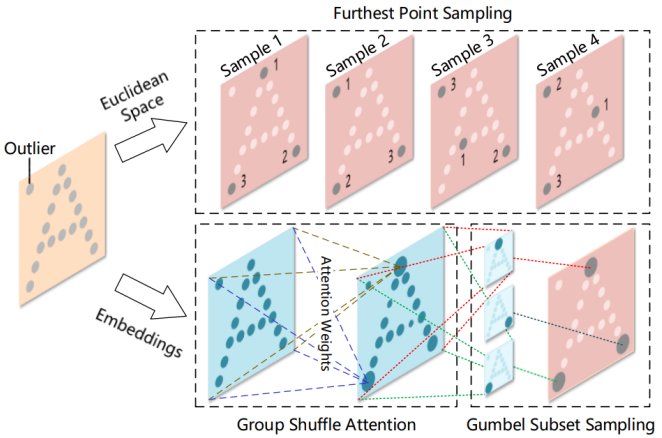
サブサンプリング手法の紹介

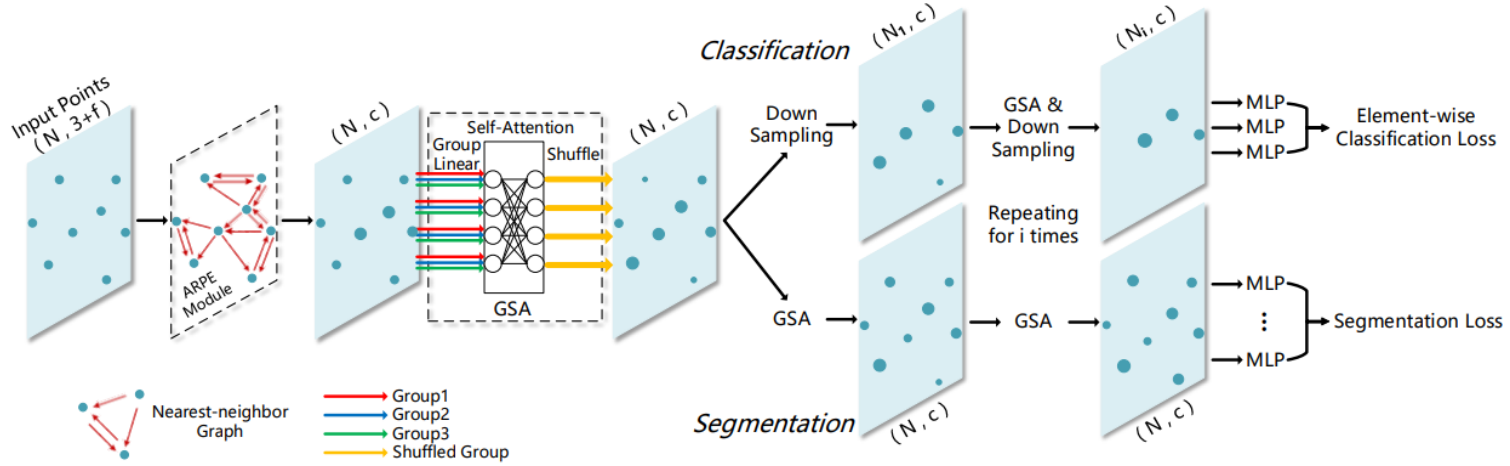
Modeling Point Clouds with Self-Attention and Gumbel Subset Sampling [J. Yang+, CVPR2019]

サンプリング方法:

Score + Gumbel-Max

1. 局所点群から各点ごとのSelection Scoreを計算
2. Gumbel-Maxで確率的にサンプリング





試すには？

- まずはFPSを試す
- FPSが遅い場合:
  - ランダムサンプリング
  - IDISS
- データが少ない・ロバストにしたい場合:
  - 学習時にランダムサンプリング
  - Scoreの導入，温度パラメータの導入など

アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - **タスクに合わせたHead**
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

タスクに合わせたHead

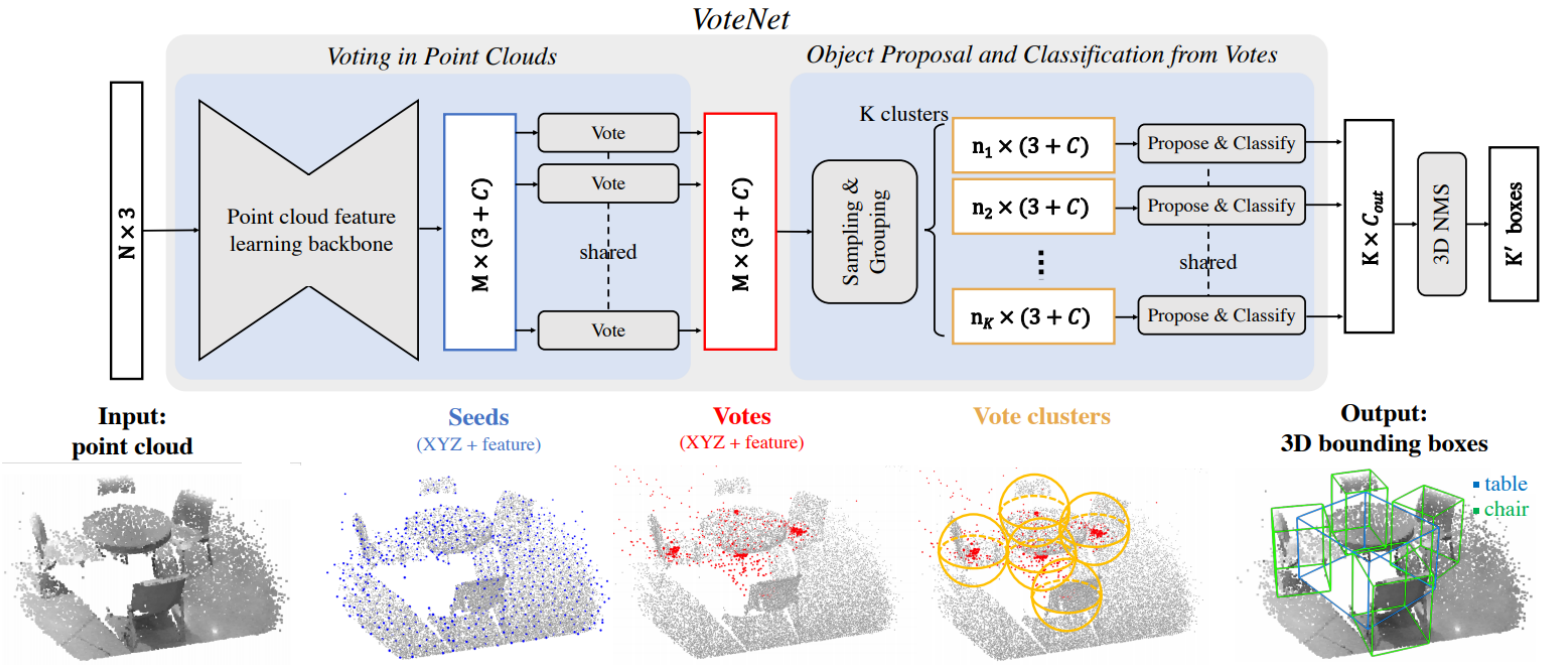
- 点群全体について出力する場合
  点群全体で集約（Max, Mean等）すればOK
- インスタンスについて出力する場合
  次スライドからVoteNetを紹介
- 各点について出力する場合
  Shared MLPなどで各点について
  分類・回帰などをすればOK
- 点群など3D形状を出力する場合
  3D形状の出力として後述

インスタンス検出

VoteNet [C. R. Qi+, ICCV2019]

物体認識＆位置姿勢推定を学習，物体クラスとバウンディングボックスを推定

サブサンプリング→ 物体座標を推定しVoting
→ K個のクラスタに分離 → Non-Max Suppression



アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - **三次元形状の出力**
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

点群の出力

三次元形状を出力する手法

- **順不同・非グリッド**な点群をどうやって出力するか
- どう比較してロスを計算するか
  - Chamfer Distance, Earth Mover's Distanceが主流
  - それ以外も提案されつつあるが普及はまだ，という印象

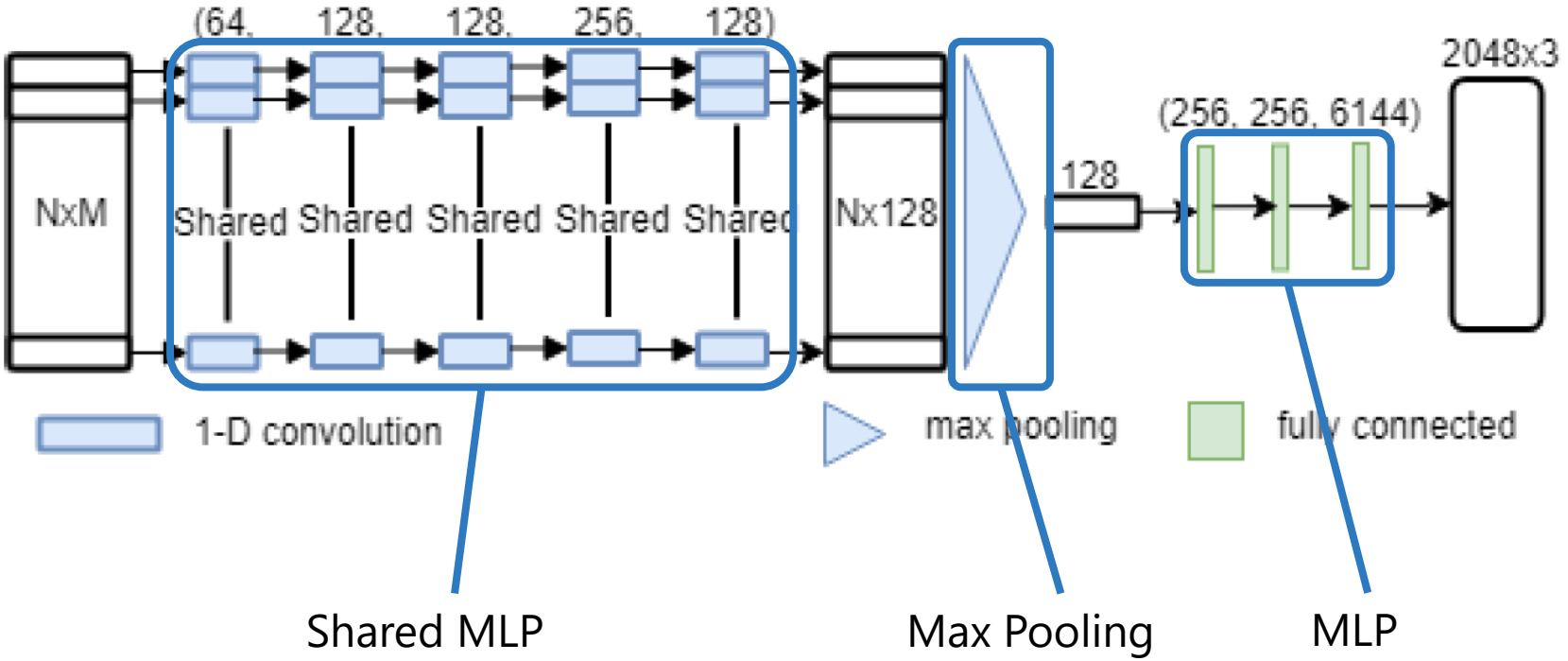
三次元点群以外での形状記述・出力が流行

- Implicit Functionによる表現
- NeRFでは3Dの教師信号が不要（今日の発表では割愛）

点群の出力手法の紹介

Data-driven Upsampling [W. Zhang+, Elsevier CAD, 2019] など

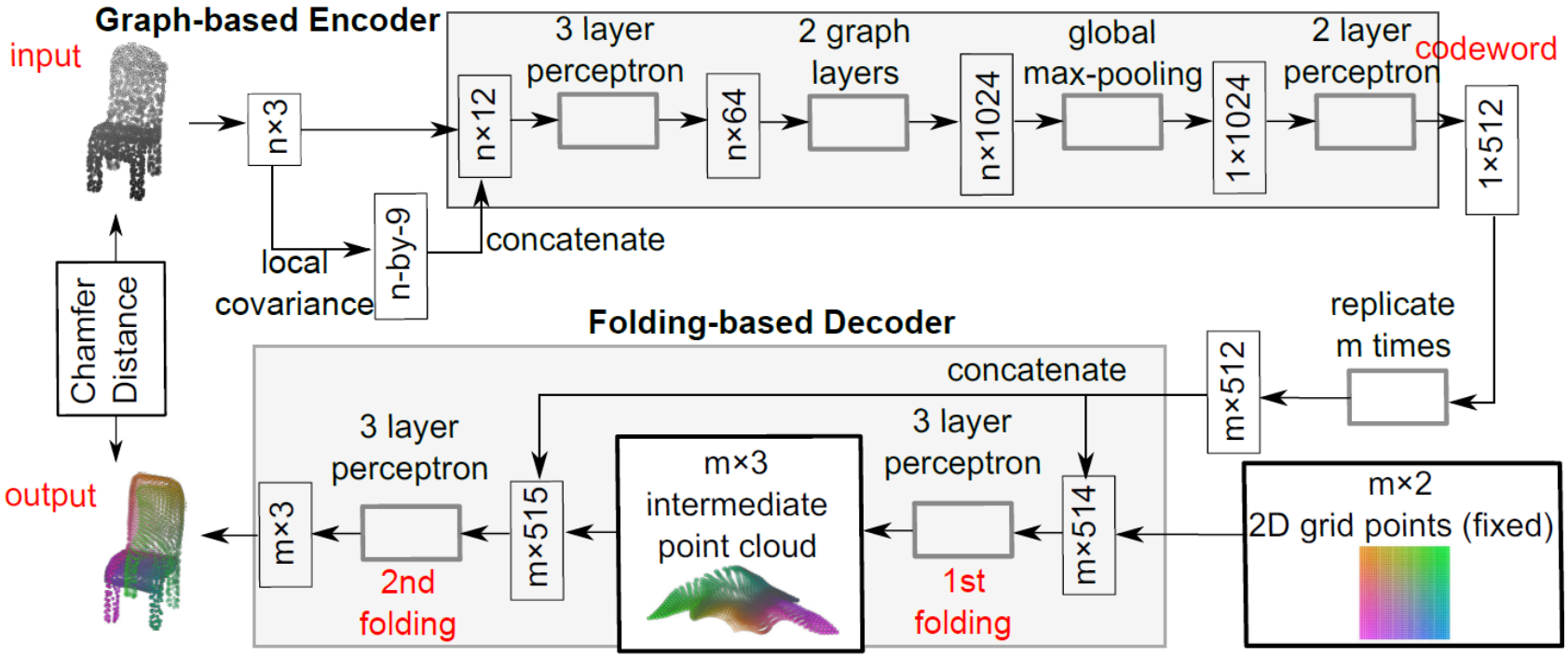
- 点群をアップサンプリングして出力
- Shared MLP+Max-pooling → MLPで点群を直接生成



点群の出力手法の紹介

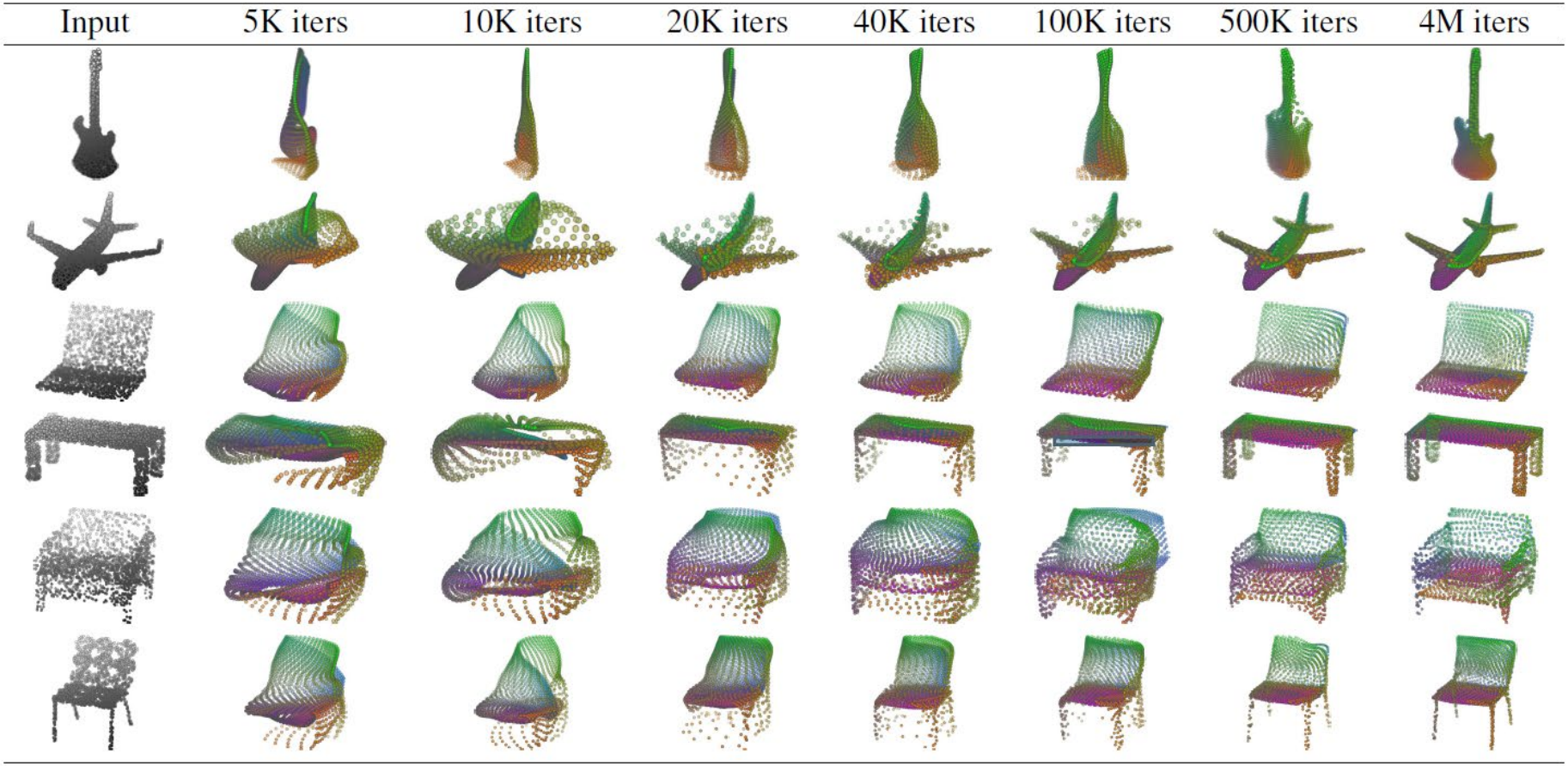
FoldingNet [Y. Yang+, CVPR2018]

- 「折りたたみ」をShared MLPでモデル化
- アイデア: 点群によって物体表面形状が記述されている
  →本質的には2D



点群の出力手法の紹介

折りたたみが徐々に学習されている

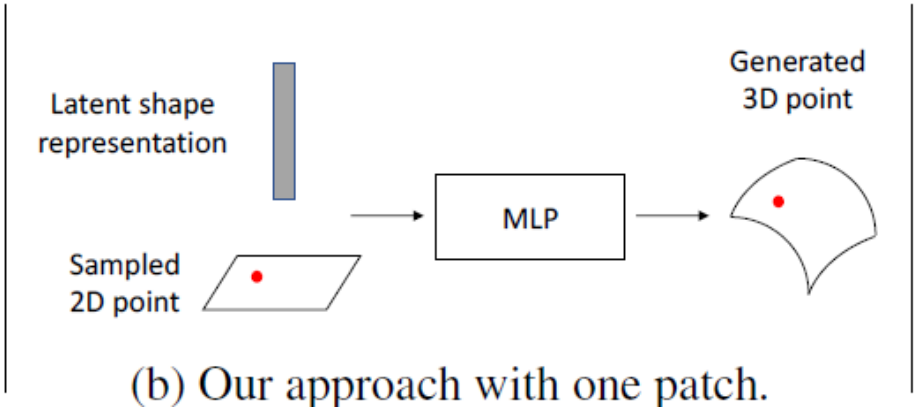


点群の出力手法の紹介

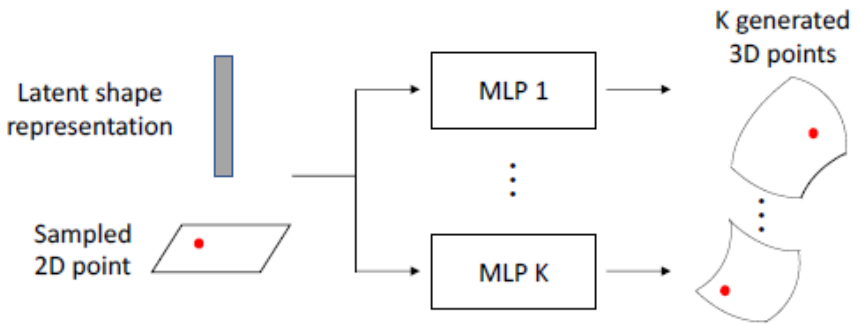
AtlasNet [T. Groueix+, CVPR2018]

単位二次元平面を三次元空間での二次元多様体にマッピングする関数φを学習

一つの関数φで三次元表面すべてを表さず，いくつかのマッピングを学習



(b) Our approach with one patch.



(c) Our approach with K patches.

単位二次元平面の点を潜在表現が示す二次元多様体上の点へマッピング

←のマッピングを複数のパッチに

点群の出力手法の紹介

アプリケーション例

- 形状補間

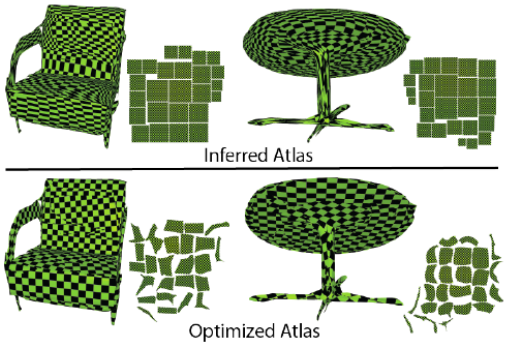
  潜在空間での線形補間により
  2形状間の三次元形状を補間

- 形状の対応関係の取得

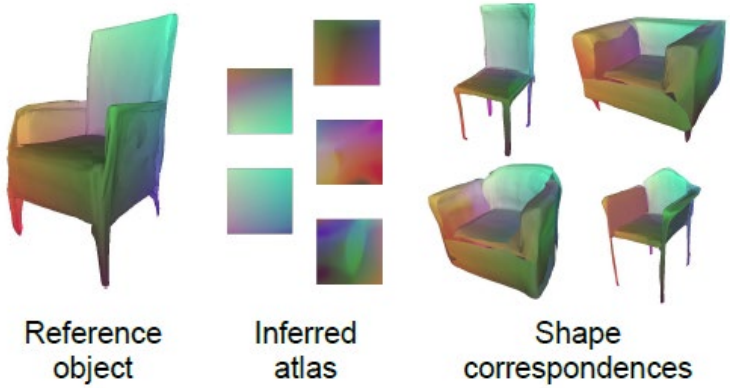
  二次元平面上で同じ点に対応する部分が，
  他の物体でも似た部分にマッピングされる

- UVマップの生成

形状補間の例



UVマップの例



形状の対応関係の取得の例

Implicit Functionによる表現

三次元形状を陽には記述せず，NNでモデル化された関数で陰的に記述

※ 今回NeRFについては割愛（3DでのSupervisionではないため）

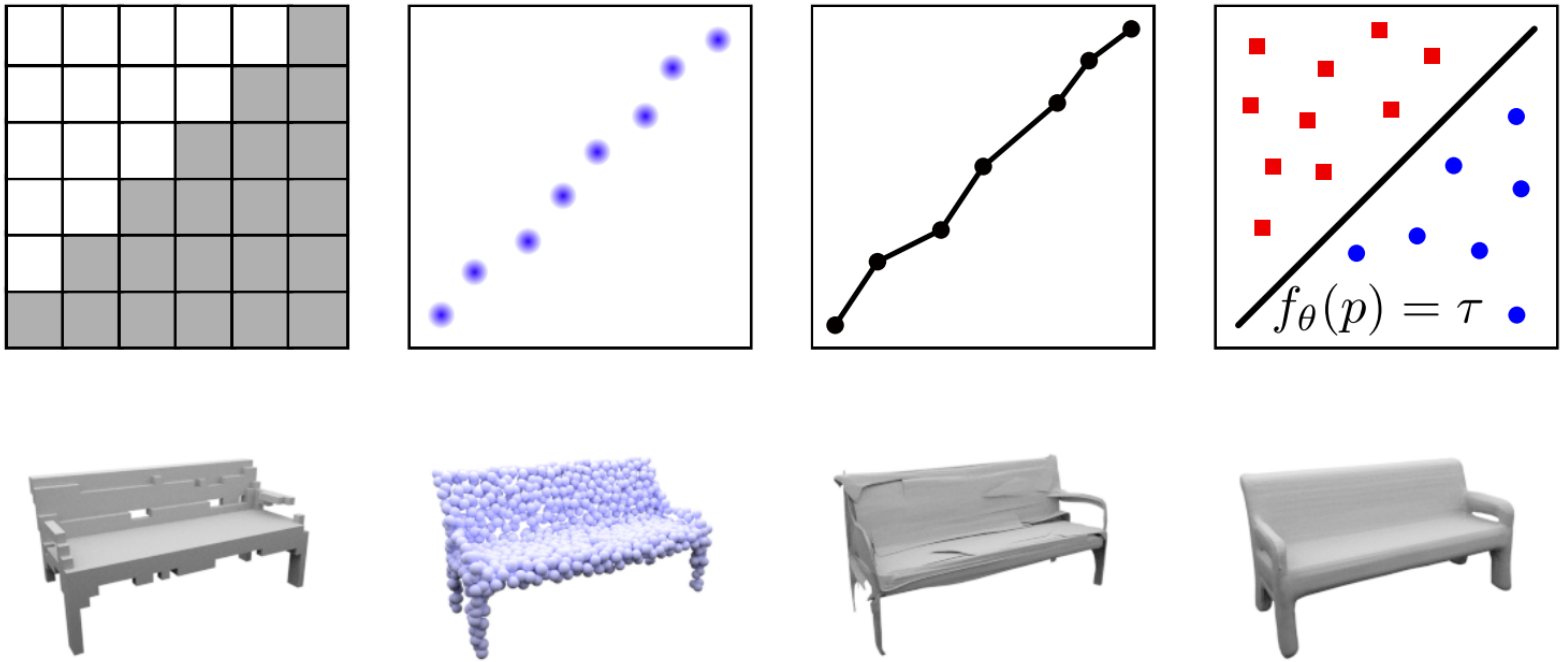
基本的なアイデア

形状をパラメータで表現して表面形状を記述，
各点をスキャンして復元

- $P = \{\boldsymbol{p}_i\}$: 三次元点群（などの，形状に対応した入力）
- $\boldsymbol{\theta}_P = g(P)$: 形状をLatent Vectorで表す
- $f(\boldsymbol{x}|\boldsymbol{\theta}_P)$: Pの表す形状のImplicit Function

Implicit Functionによる表現

Occupancy Networks [L. Mescheder+, CVPR2019]
ニューラルネットワークの識別境界で三次元形状を記述

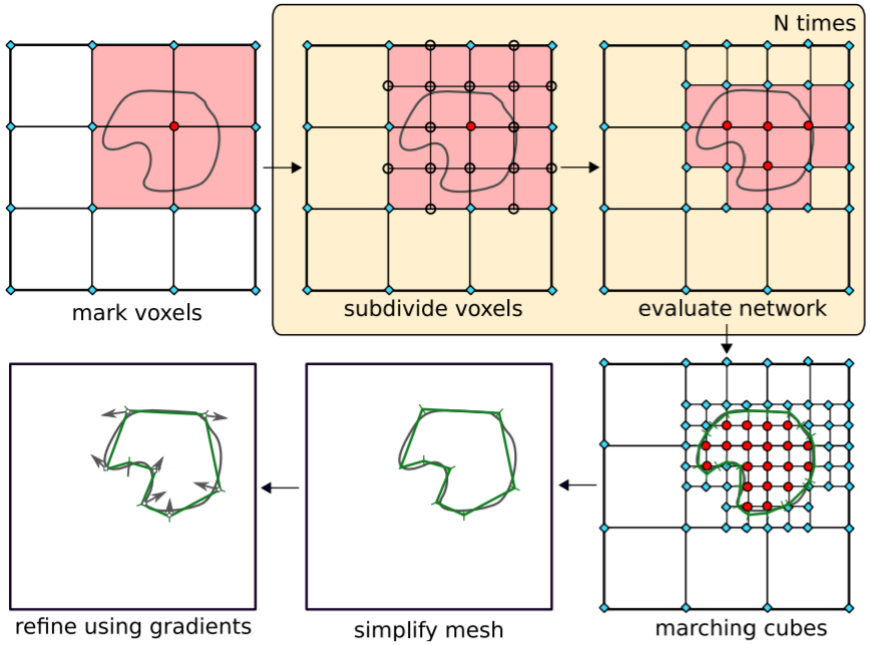


(a) Voxel (b) Point (c) Mesh (d) Ours

Implicit Functionによる表現

Occupancy Networks [L. Mescheder+, CVPR2019]

- エンコーダーは様々なものが使える
  - 2D画像
  - PointNet
  - 3D CNN （ボクセルベース）
- 陰的な表現なので出力手法が必要
  - 標準的にはMarching Cubes法が用いられる
  - この論文では Multiresolution IsoSurface Extractionを提案



Implicit Functionによる表現

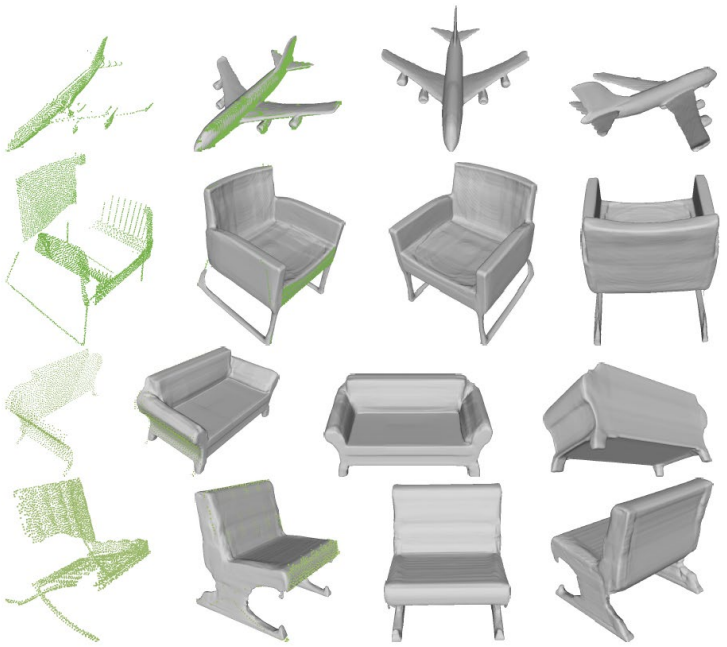
DeepSDF [J. J. Park+, CVPR2019]

形状をコード化して形状ごとのSigned Distance Function (SDF)を直接学習

シンプルなモデルで三次元形状が記述できる

Signed Distance Function

$$f(\boldsymbol{x})\begin{cases} > 0 & \text{（物体の外側）} \\ = 0 & \text{（物体の表面）} \\ < 0 & \text{（物体の内側）} \end{cases}$$



DeepSDFによる全周形状の補完の例

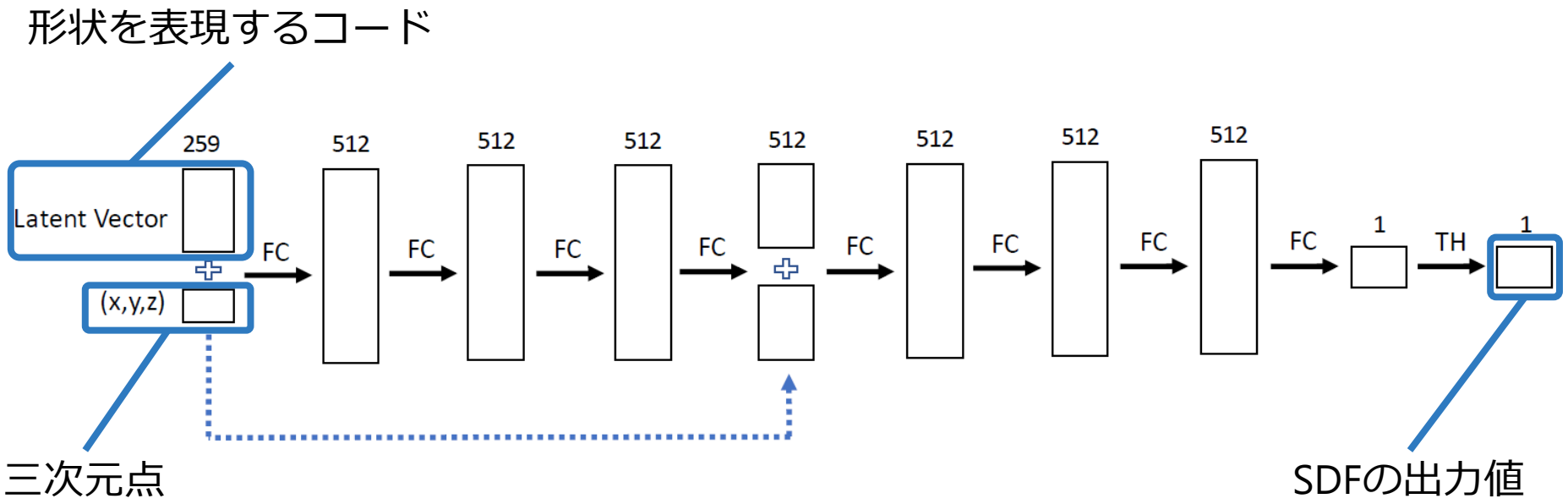
Implicit Functionによる表現

DeepSDF [J. J. Park+, CVPR2019]

ネットワークの構成:

SDF自体を学習するため，ネットワークの構造はシンプル

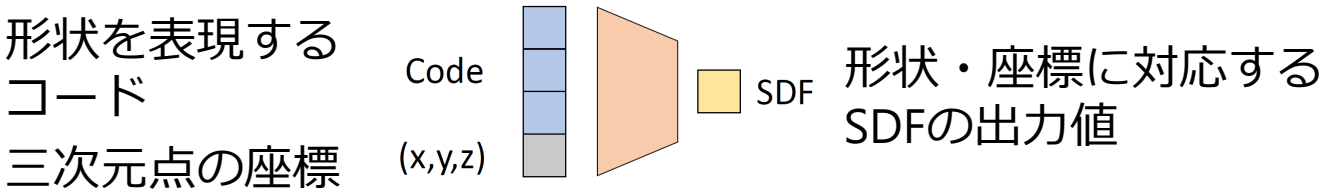
基本的には単純なMLP



Implicit Functionによる表現

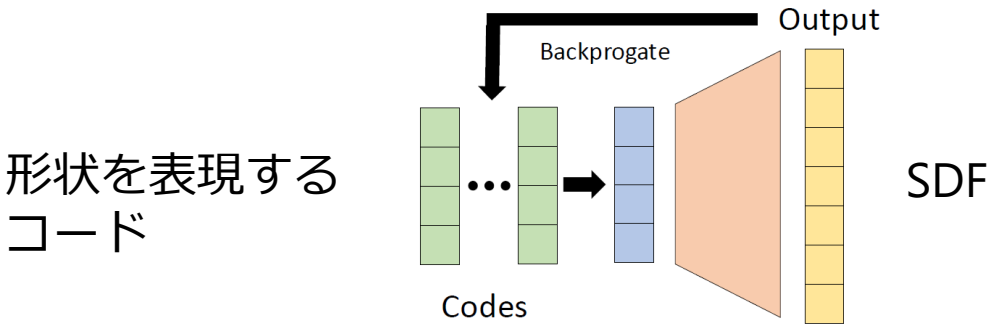
DeepSDF [J. J. Park+, CVPR2019]

三次元形状をコードで表現し，そのモデルに対するSDFの値を各点について直接学習



Auto-decoder

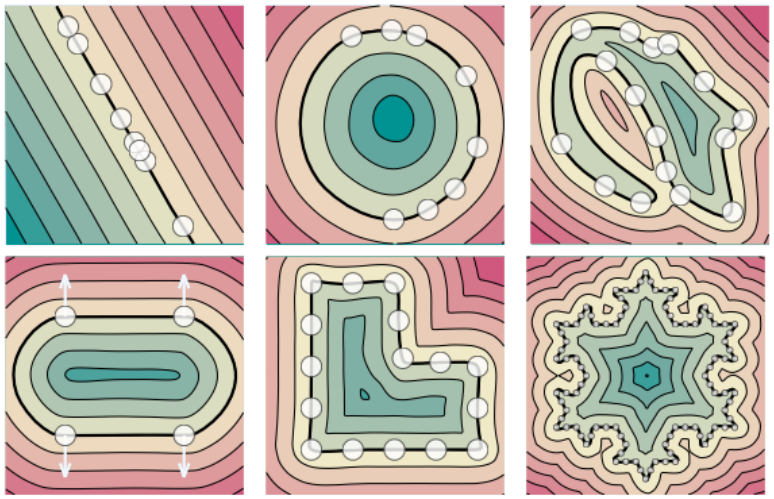
- エンコーダーを置かず，直接コードを学習
- 推論時はMAP推定で形状からコードに変換できる

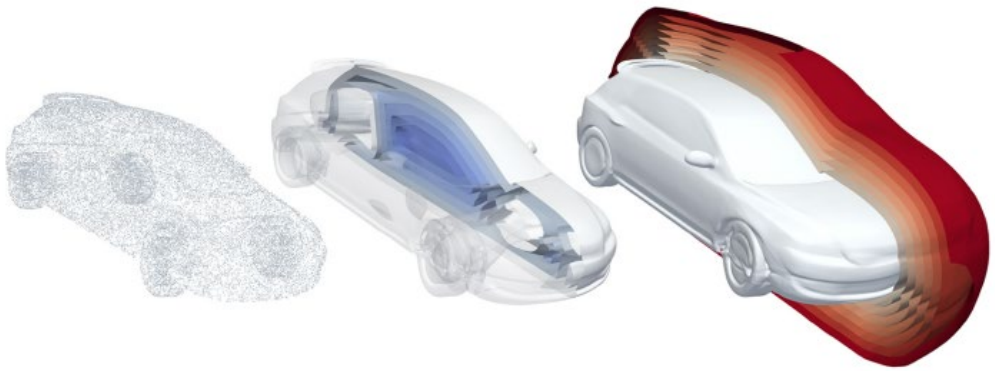


表面点群のみから学習

Implicit Geometric Regularization [A.Gropp+, ICML2020]

- 表面点群（＝Implicit Functionのゼロ面上の点）からImplicit FunctionによりSDFを学習
- 適切な初期化と空間中での勾配に関する制約（SDFの勾配があらゆる点で1になる）を用いて学習





試すには？

まずは比較的単純な方法で試す

- （Voxelによる形状を3D CNNで生成）
- MLPで直接生成
- FoldingNet

高解像度な形状が必要な場合:
Implicit Functionによる表現を検討

体感ではだいぶ学習が難しい＆時間がかかる

アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - **事前学習**
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- ライブラリなど

事前学習

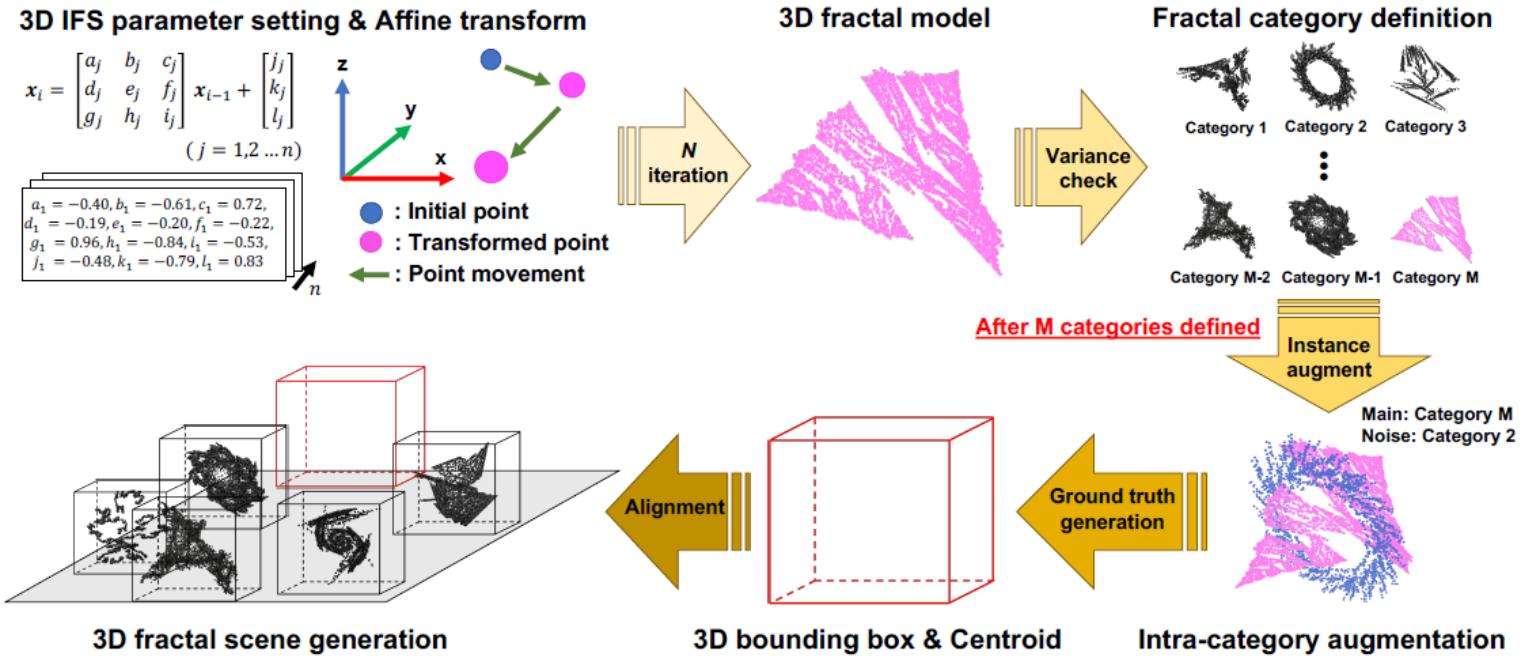
点群は**大規模データセットの用意が難しい**ため
事前学習が効く可能性が高い

大規模なラベル付きデータセットを必要としない
自己教師による手法や自動生成可能な手法に期待
まだ発展途上，組み合わせて使うことも可能なはず

- Natural 3D Structures [R. Yamada+, CVPR2022]
  フラクタルを利用したモデル生成＆配置を自動生成し物体検出を事前学習
- Implicit Autoencoder [S. Yan+, arXiv, 2022]
  Implicit Functionによるデコーダーと組み合わせてオートエンコーダー
- DepthContrast [Z. Zhang+, ICCV2021]
  モーダルの異なる3Dデータを利用して対照学習

事前学習

- Natural 3D Structures [R. Yamada+, CVPR2022]
  1. 3D Iterated Function System (IFS) でモデル生成
  2. 分散に基づいてカテゴリ定義
  3. 複数モデルを重ね合わせてデータ拡張
  4. 3Dで配置しシーン作成



アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- **点群を扱うニューラルネットワーク**
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- **アプリケーションの例**
  - **典型的なアプリケーション**
  - 点群位置合わせ
- ライブラリなど

アプリケーションの例

ベンチマーク的なアプリケーション

- クラス分類
- （セマンティック／インスタンス）セグメンテーション

実利用寄りなアプリケーション

- 物体検出・物体認識

  VoxelNet [Y. Zhou+, CVPR2018]，Frustum PointNet [C. R. Qi+, CVPR2018]など
- 点群の位置合わせ（マッチングベース）

  3DMatch [A. Zeng+, CVPR2017]，3D Feat-Net [Z. J. Yew+, ECCV2018]など
- 点群の位置合わせ（剛体変換推定）

  IT-Net [W. Yuan+, arXiv, 2018]，PointNetLK [Y. Aoki+, CVPR2019]など
- 点群の位置合わせ（教師なし非剛体）
  - CorrNet3D [Y. Zeng+, CVPR2021]，ESFW [Y. Rintato+, arXiv, 2022]
- アップサンプリング

  PU-Net [L. Yu+, CVPR2018]，Progressive Upsampling [Yuan+, 3DV2018]など
- 形状補完

  Under Weak Supervision [D. Stutz+, CVPR2018]，Latent Optimization [S. Gurumurthy+, arXiv, 2018]など
- その他

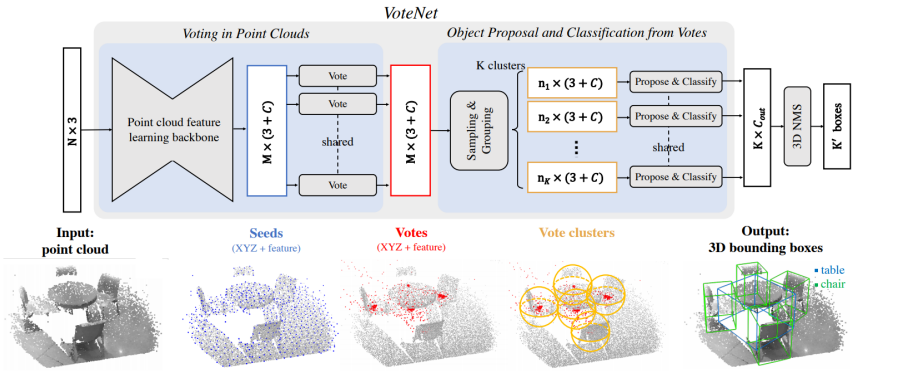
  ノイズ除去 [M. J. Rakotosaona+, arXiv, 2019]，手の姿勢推定 [L. Ge+, ECCV2018]，
  把持姿勢の評価 [H. Liang+, ICRA2019] など

アプリケーションの例

物体検出

VoteNet [C. R. Qi+, ICCV2019]

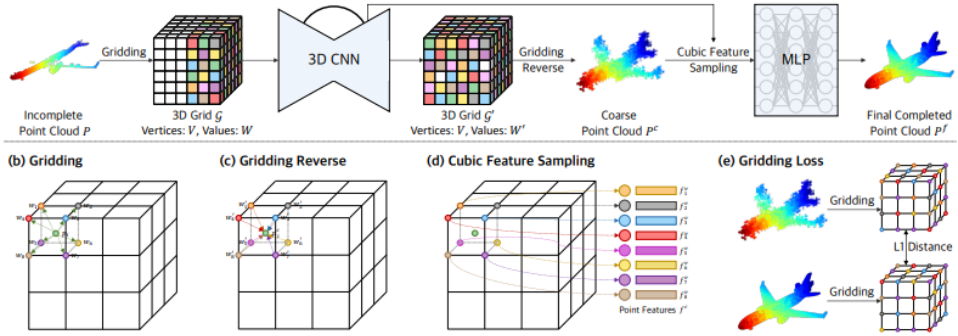
Votingを用いて三次元点群から
物体中心を推定，クラスタリングし
物体検出・BB推定



形状補完

GRNet [H. Xie+, ECCV2020]

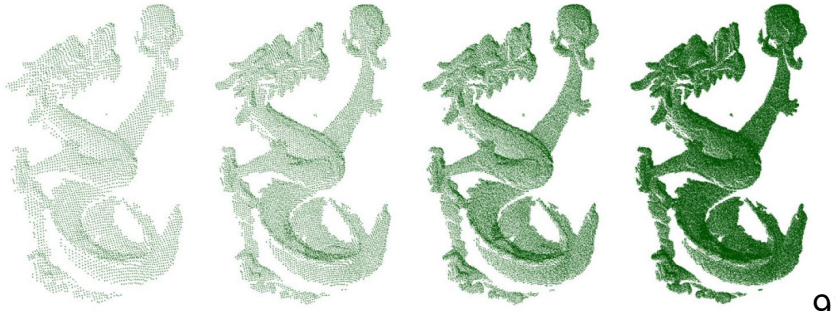
ボクセル畳み込み＆Refineで
全周点群の補完



アップサンプリング

Progressive Upsampling
[W. Yifan+, CVPR2019]

パッチベースのアップサンプリング



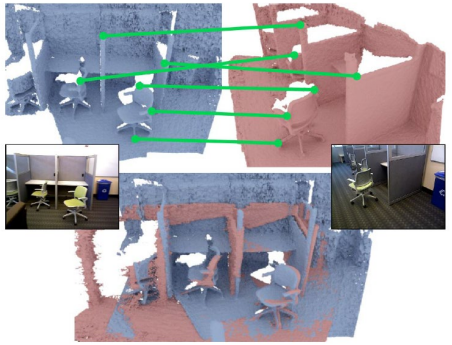
アプリケーションの例

点群の位置合わせ

- マッチングベース

  3DMatch [A. Zeng+, CVPR2017]

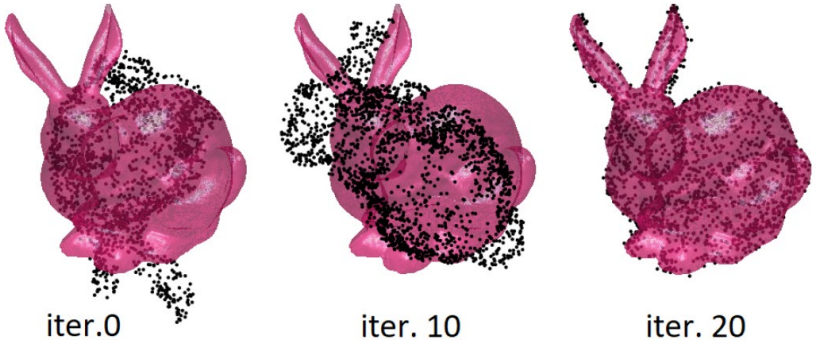
  位置合わせのための
  局所点群を記述する特徴量を学習



- 剛体変換推定

  PointNetLK [Y. Aoki+, CVPR2019]

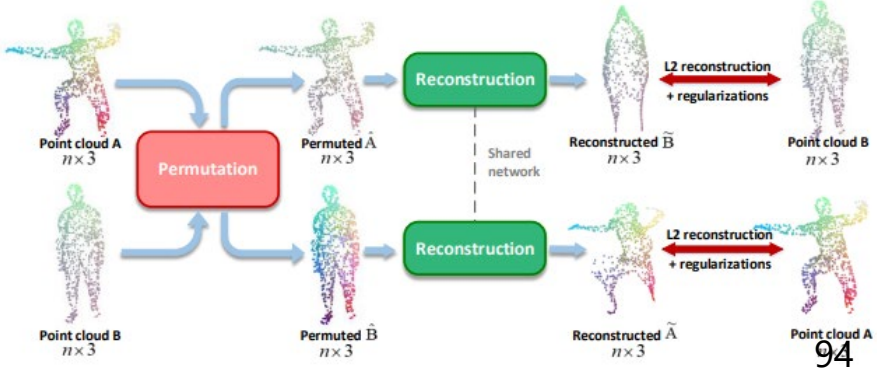
  Lucas-Kanade法とPointNetを
  組み合わせて点群同士の位置合わせ



- 教師なし非剛体位置合わせ

  CorrNet3D [Y. Zeng+, CVPR2021]

  ESFW [R. Yanagi+, arXiv]

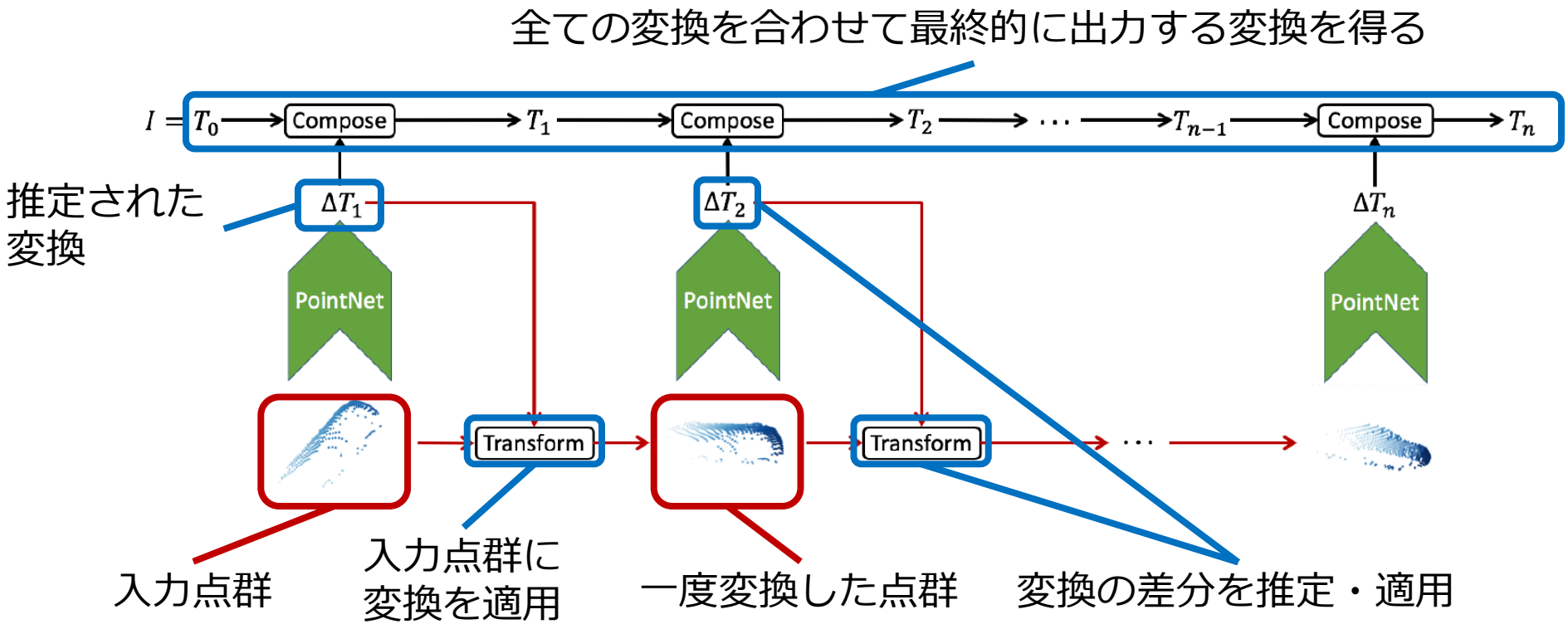


アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- 点群を扱うニューラルネットワーク
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- **アプリケーションの例**
  - 典型的なアプリケーション
  - **点群位置合わせ**
- ライブラリなど

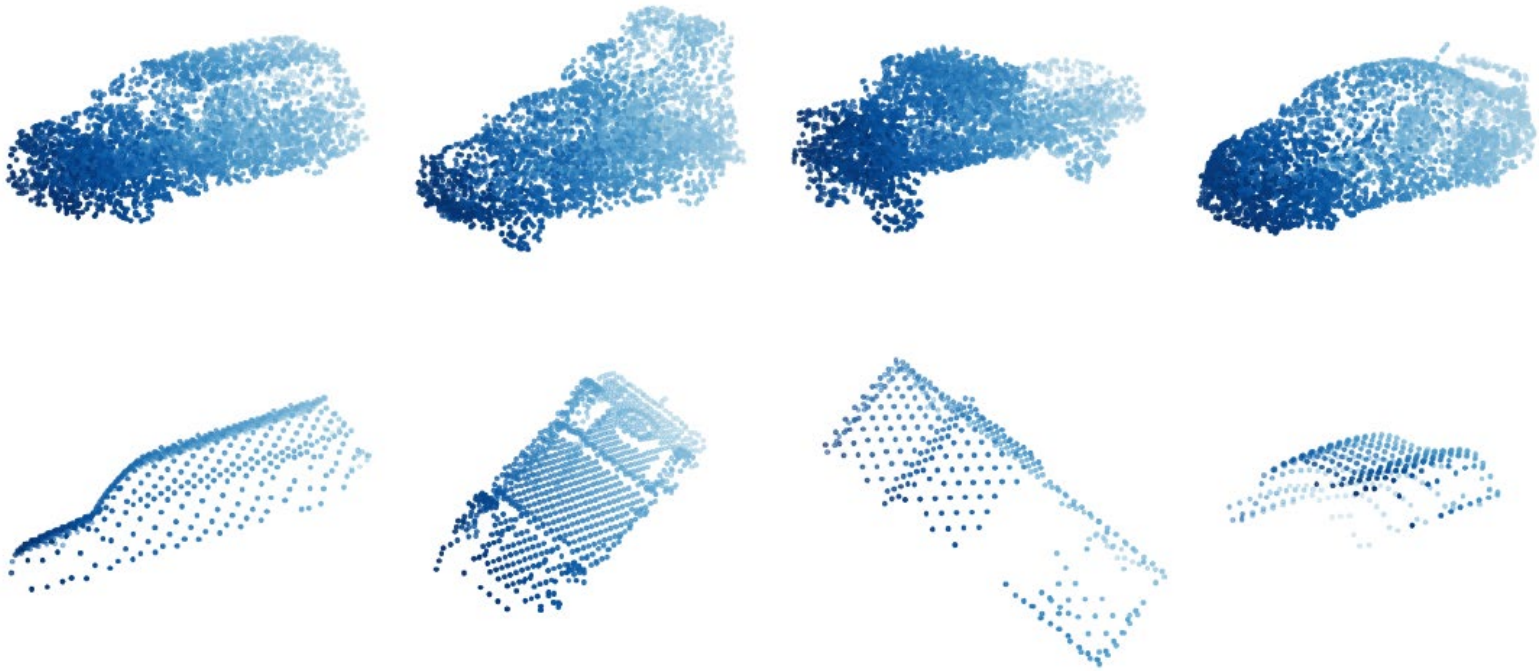
点群位置合わせ（剛体変換）

IT-Net [W. Yuan+, arXiv:1811.11209]
3D剛体変換を学習，回転をそろえるための
Iterative Transformer Network (IT-Net)を提案



点群位置合わせ（剛体変換）

IT-Net [W. Yuan+, arXiv:1811.11209]
　片面点群・センサ座標系を想定したデータセットで学習

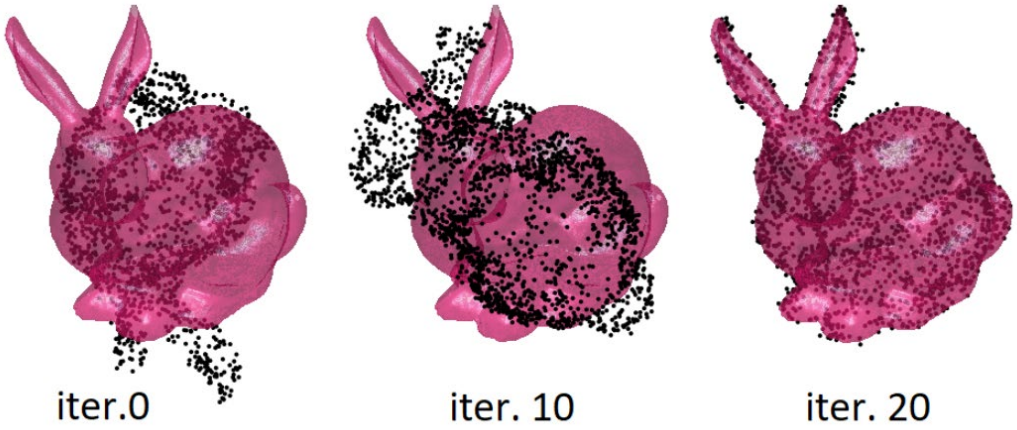


上: 既存のモデル 下: 本論文のデータセット
観測が片面かつ回転が正規化されていない

点群位置合わせ（剛体変換）

PointNetLK
[Y. Aoki+, CVPR2019]

Lucas-Kanade法に
PointNetを組み合わせ，
点群位置合わせ



位置合わせの例

剛体変換: 三次元ユークリッド空間でのSE(3)の要素

6次元のTwist Parametersを用いる
SE(3)の指数写像のGeneratorの重み付け和を求め，指数関数で
変換した行列として剛体変換を記述

$$\mathbf{G} = \exp\left(\sum_i \xi_i \mathbf{T}_i\right) \qquad \boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_6)^T$$

剛体変換　パラメータベクトル　SE(3)についてのGenerator

点群位置合わせ（剛体変換）

PointNetLK [Y. Aoki+, CVPR2019]

ϕをPointNetによる特徴ベクトルへの変換とする
ICを導入することを前提にすると，

$$\phi(\mathbf{P}_{\mathcal{S}}) = \phi(\mathbf{G}^{-1} \cdot \mathbf{P}_{\mathcal{T}})$$

変換元の点群　求める剛体変換の逆変換　変換先の点群

Inverse Compositionalで考え，$\mathbf{G}^{-1}$について解く
変換先のパラメータ周りで一次近似する

$$\phi(\mathbf{P}_{\mathcal{S}}) = \phi(\mathbf{P}_{\mathcal{T}}) + \frac{\partial}{\partial \boldsymbol{\xi}} \left[\phi(\mathbf{G}^{-1} \cdot \mathbf{P}_{\mathcal{T}})\right] \boldsymbol{\xi}$$

パラメータに対するヤコビアン
以降$\mathbf{J}$と表記

パラメータ

点群位置合わせ（剛体変換）

PointNetLK [Y. Aoki+, CVPR2019]

パラメータに対するPointNetの微分を考えるのは
非常に難しい → 数値的にヤコビアンを計算
ヤコビアン$\mathbf{J}$の各列$\mathbf{J}_i$をパラメータの微小変動で近似

$$\mathbf{J}_i = \frac{\phi(\exp(-t_i \mathbf{T}_i) \cdot \mathbf{P}_\mathcal{T}) - \phi(\mathbf{P}_\mathcal{T})}{t_i}$$

$\mathbf{T}$に沿った
微小な剛体変換

微小な剛体変換後の点群に
対するPointNetの出力

元の点群に対する
点群のPointNetの出力

ヤコビアンの疑似逆行列によりパラメータを更新

$$\boldsymbol{\xi} = \mathbf{J}^{+} \left[\phi(\mathbf{P}_\mathcal{S}) - \phi(\mathbf{P}_\mathcal{T}) \right]$$

上で求めた近似ヤコビアンの疑似逆行列

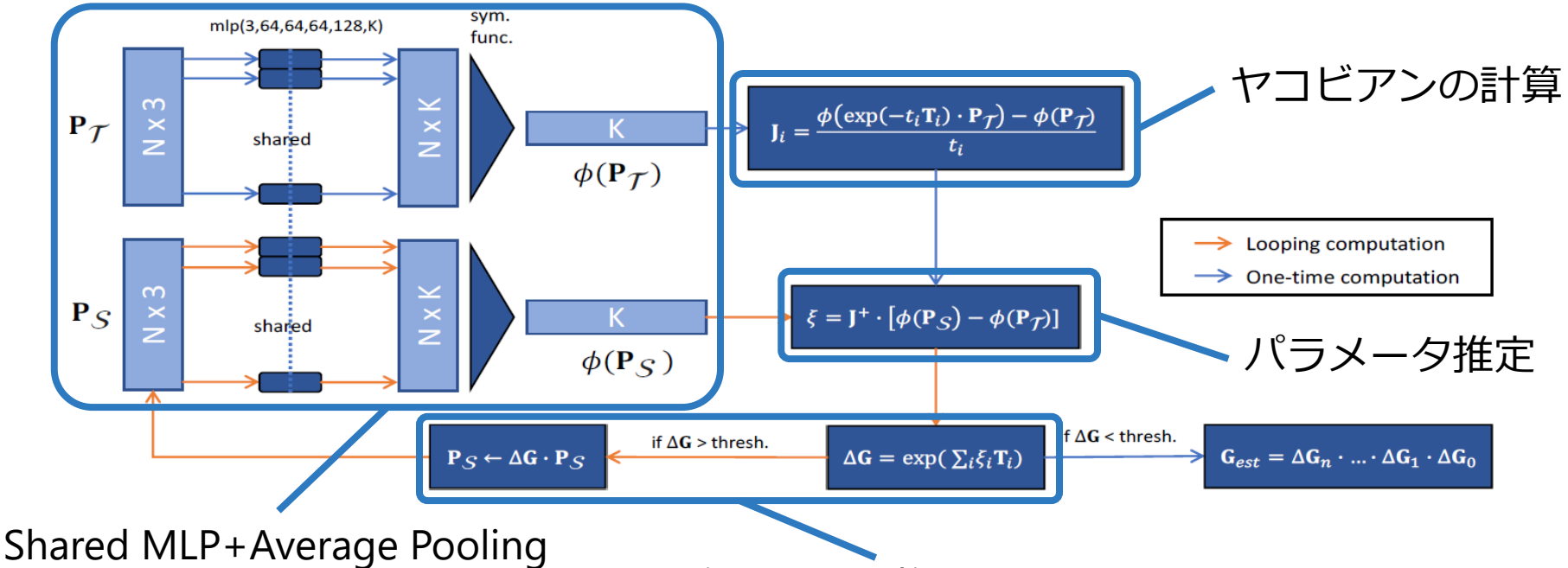
点群位置合わせ（剛体変換）

PointNetLK [Y. Aoki+, CVPR2019]

ネットワークの構造

PointNetに近い構造，T-Netを取り除いている

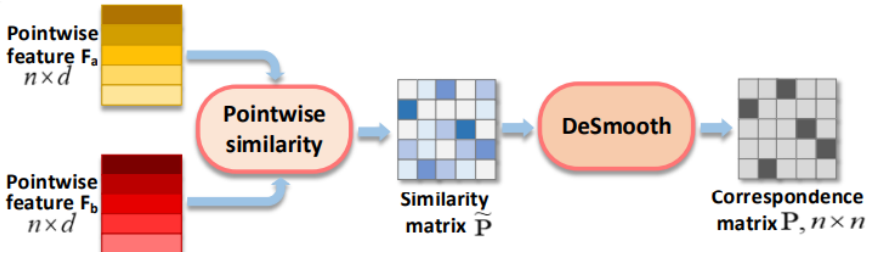
行列対数関数を計算しなくて良いように，順変換→逆変換した行列と単位行列を比較



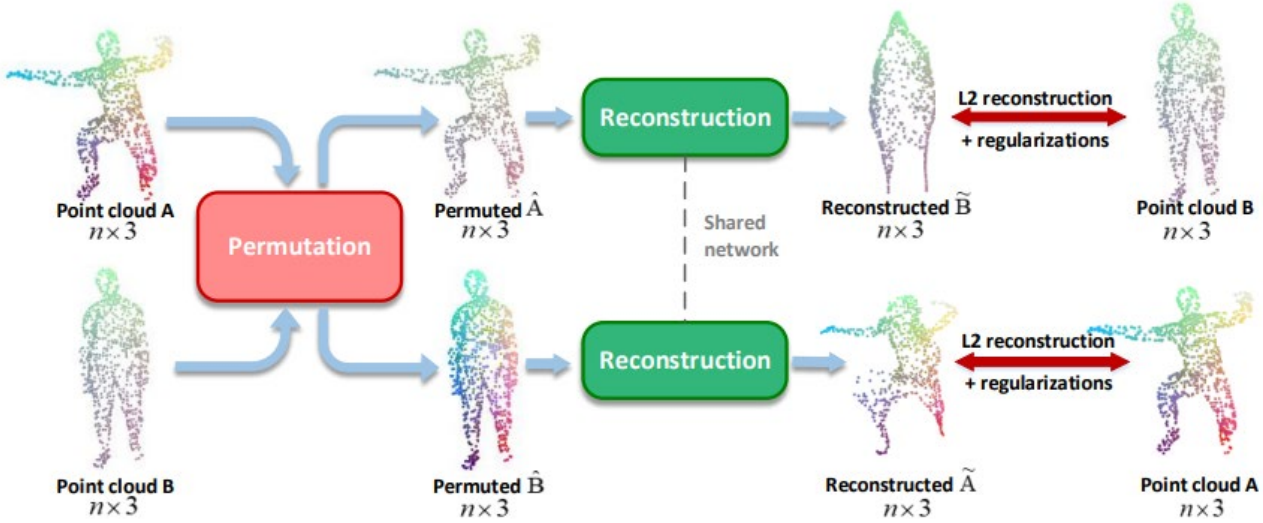
点群位置合わせ（非剛体変換）

CorrNet3D [Y. Zeng+, CVPR2021]

- 非剛体な変形をする点群間の対応を教師なしで学習
  - 局所点群の特徴から形状の対応を推定
- 変形した点群を入力し対応を推定，並べ替え変形してみて比較



局所特徴量のSimilarityから対応推定



アウトライン

- 三次元形状のデータ形式と計測
  - さまざまなデータ形式
  - 三次元計測
- 点群を扱うニューラルネットワーク
  - 点群を扱う難しさ
  - 典型的なタスク・ネットワーク構成
  - PointNetの紹介
  - 点群の畳み込み・Backbone
  - タスクに合わせたHead
  - 三次元形状の出力
  - 事前学習
- アプリケーションの例
  - 典型的なアプリケーション
  - 点群位置合わせ
- **ライブラリなど**

ライブラリなど

現状でのオススメ：**PyTorch Geometric**

- グラフNN的な枠組みで点群処理が書ける
- 有名なネットワーク・よく使われる手法が実装済み

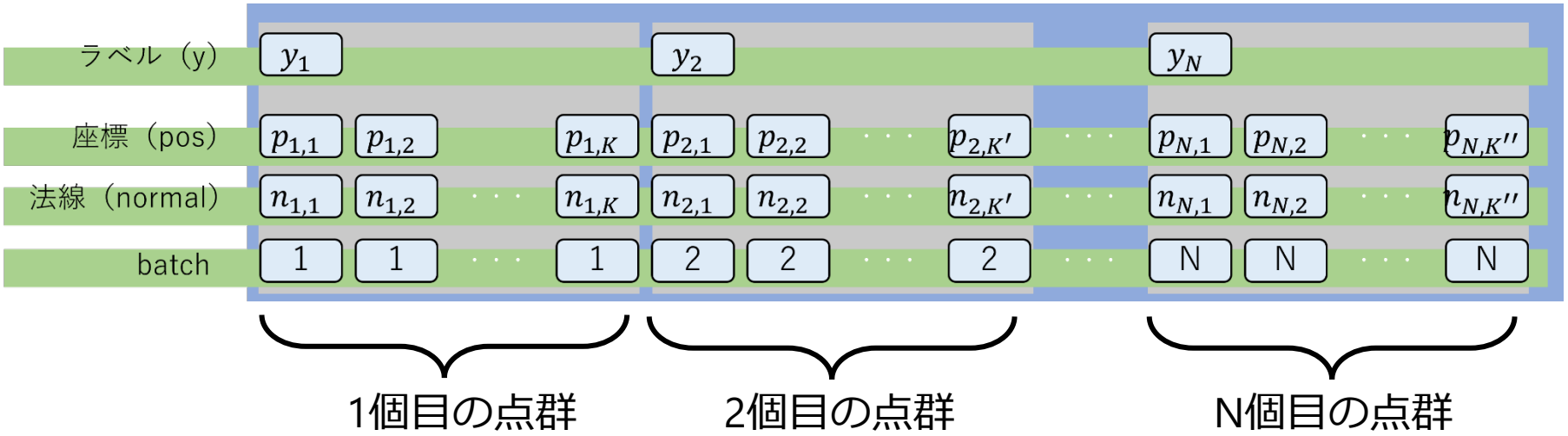
その他のライブラリ

- PyTorch3D, TensorFlow Graphics
  - メッシュの最適化・微分可能レンダリング等に強い
- Kaolin
  - 3D全般・Omniverseに組み込まれるらしい？
- Minkowski Engine, TorchSparse
  - スパーステンソルの扱いに強い
- MATLABも対応しつつあるらしい

PyTorch Geometricの紹介

PyTorch Geometricにおけるミニバッチ

- Tensorとしてはすべて結合している
- どの点群（グラフ）に属するかをbatchで指定
  （PyTorch Scatterを利用している）
- Shared MLP等の要素ごとの処理はそのまま動く
- 集約やkNN等をグラフ単位で処理できる関数を用意



まとめ

- 三次元点群に深層学習を適用する際の
  - キーとなるアイデア：Symmetric Function
  - 典型的なネットワーク構成：Backbone+Head
  - Backboneで使われる点群畳み込み
  - 点群（・三次元形状）を出力する際のアプローチ

  について解説
- アプリケーション例を紹介

参考文献 (1/3)

- Eman Ahmed, Alexandre Saint, Abd El Rahman Shabayek, Kseniya Cherenkova, Rig Das, Gleb Gusev, Djamila Aouada, Bjorn Ottersten. A survey on Deep Learning Advances on Different 3D Data Representations. arXiv:1808.01462.
- Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm & Marc Pollefeys. Pixelwise View Selection for Unstructured Multi-View Stereo, ECCV2016.
- Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, Daniele Panozzo. ABC: A Big CAD Model Dataset For Geometric Deep Learning, CVPR2019.
- Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR2017.
- Charles R. Qi, Or Litany, Kaiming He, Leonidas Guibas. Deep Hough Voting for 3D Object Detection in Point Clouds. ICCV2019.
- Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, Alexander Smola. Deep Sets. NIPS2017.
- Max Jaderberg, Karen Simonyan, Andrew Zisserman, Koray Kavukcuoglu. Spatial Transformer Networks. NIPS2015.
- Hongxin Lin, Zelin Xiao, Yang Tan, Hongyang Chao, Shengyong Ding. Justlookup: One Millisecond Deep Feature Extraction for Point Clouds By Lookup Tables. ICME2019.
- Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, Raquel Urtasun. Deep Parametric Continuous Convolutional Neural Networks. CVPR2018.
- Binh-Son Hua, Minh-Khoi Tran, Sai-Kit Yeung. Pointwise Convolutional Neural Networks. CVPR2018.
- Fabian Groh, Patrick Wieschollek, Hendrik P.A. Lensch. Flex-Convolution (Million-Scale Point-Cloud Learning Beyond Grid-Worlds). ACCV2018.
- Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, Qian-Yi Zhou. Tangent Convolutions for Dense Prediction in 3D. CVPR2018.
- Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, Oscar Beijbom. PointPillars: Fast Encoders for Object Detection from Point Clouds. CVPR2019.
- Charles Ruizhongtai Qi, Li Yi, Hao Su, Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. NIPS2017.
- Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, Justin M. Solomon. Dynamic Graph CNN for Learning on Point Clouds. ACM ToG, 2019.
- Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, Yu Qiao. SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters. ECCV2019.
- Hengshuang Zhao, Li Jiang, Chi-Wing Fu, Jiaya Jia. PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing. CVPR2019.
- Artem Komarichev, Zichun Zhong, Jing Hua. A-CNN: Annularly Convolutional Neural Networks on Point Clouds. CVPR2019.
- Zhiyuan Zhang, Binh-Son Hua, Sai-Kit Yeung. ShellNet: Efficient Point Cloud Convolutional Neural Networks Using Concentric Shells Statistics. ICCV2019.
- Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, Yun Fu. Rethinking Network Design and Local Geometry in Point Cloud: A Simple Residual MLP Framework. ICLR2022.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need. NIPS2017.

参考文献 (2/3)

- Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H.S. Torr, Vladlen Koltun. Point Transformer. ICCV2021.
- Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, Gao Huang. 3D Object Detection With Pointformer. CVPR2021.
- Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, Alexey Dosovitskiy. MLP-Mixer: An all-MLP Architecture for Vision. NeurIPS2021.
- Jaesung Choe, Chunghyun Park, Francois Rameau, Jaesik Park, In So Kweon. PointMixer: MLP-Mixer for Point Cloud Understanding. arXiv:2111.11187.
- Jiaxin Li, Gim Hee Lee. USIP: Unsupervised Stable Interest Point Detection From 3D Point Clouds. ICCV2019.
- Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, Qi Tian. Modeling Point Clouds with Self-Attention and Gumbel Subset Sampling. CVPR2019.
- Wentai Zhang, Haoliang Jiang, Zhangsihao Yang, Soji Yamakawa, Kenji Shimada, Levent Burak Kara. Data-driven Upsampling of Point Clouds. Elsevier CAD, 2019.
- Yaoqing Yang, Chen Feng, Yiru Shen, Dong Tian. FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation. CVPR2018.
- Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. CVPR2018.
- Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV2020.
- Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, Andreas Geiger. Occupancy Networks: Learning 3D Reconstruction in Function Space. CVPR2019.
- Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. CVPR2019.
- Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, Yaron Lipman. Implicit Geometric Regularization for Learning Shapes. ICML2020.
- Ryosuke Yamada, Hirokatsu Kataoka, Naoya Chiba, Yukiyasu Domae, Tetsuya Ogata. Point Cloud Pre-training with Natural 3D Structures. CVPR2022.
- Siming Yan, Zhenpei Yang, Haoxiang Li, Li Guan, Hao Kang, Gang Hua, Qixing Huang. Implicit Autoencoder for Point Cloud Self-supervised Representation Learning. arXiv:2201.00785.
- Zaiwei Zhang, Rohit Girdhar, Armand Joulin, Ishan Misra. Self-Supervised Pretraining of 3D Features on Any Point-Cloud. ICCV2021.
- Yin Zhou, Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. CVPR2018.
- Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, Leonidas J. Guibas. Frustum PointNets for 3D Object Detection from RGB-D Data. CVPR2018.
- Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, Thomas Funkhouser. 3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions. CVPR2017.

参考文献 (3/3)

- Zi Jian Yew, Gim Hee Lee. 3DFeat-Net: Weakly Supervised Local 3D Features for Point Cloud Registration. ECCV2018.
- Wentao Yuan, David Held, Christoph Mertz, Martial Hebert. Iterative Transformer Network for 3D Point Cloud. arXiv:1811.11209.
- Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, Simon Lucey. PointNetLK: Robust & Efficient Point Cloud Registration using PointNet. CVPR2019.
- Yiming Zeng, Yue Qian, Zhiyu Zhu, Junhui Hou, Hui Yuan, Ying He. CorrNet3D: Unsupervised End-to-End Learning of Dense Correspondence for 3D Point Clouds. CVPR2021.
- Rintaro Yanagi, Atsushi Hashimoto, Shusaku Sone, Naoya Chiba, Jiaxin Ma, Yoshitaka Ushiku. Edge-Selective Feature Weaving for Point Cloud Matching. arXiv:2202.02149.
- Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, Pheng-Ann Heng. PU-Net: Point Cloud Upsampling Network. CVPR2018.
- Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, Olga Sorkine-Hornung. Patch-based Progressive 3D Point Set Upsampling. CVPR2019.
- David Stutz, Andreas Geiger. Learning 3D Shape Completion From Laser Scan Data With Weak Supervision. CVPR2018.
- Swaminathan Gurumurthy, Shubham Agrawal. High Fidelity Semantic Shape Completion for Point Clouds using Latent Optimization. arXiv:1807.03407.
- Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J. Mitra, Maks Ovsjanikov. PointCleanNet: Learning to Denoise and Remove Outliers from Dense Point Clouds. arXiv:1901.01060.
- L. Ge, Z. Ren, J. Yuan. Point-to-Point Regression PointNet for 3D Hand Pose Estimation. ECCV2018.
- Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun, Jianwei Zhang. PointNetGPD: Detecting Grasp Configurations from Point Sets. ICRA2019.
- Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, Wenxiu Sun. GRNet: Gridding Residual Network for Dense Point Cloud Completion. ECCV2020.
- Yiming Zeng, Yue Qian, Zhiyu Zhu, Junhui Hou, Hui Yuan, Ying He. CorrNet3D: Unsupervised End-to-End Learning of Dense Correspondence for 3D Point Clouds. CVPR2021.