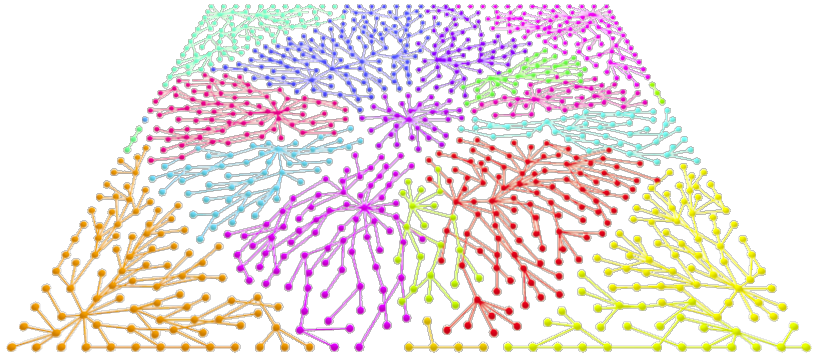
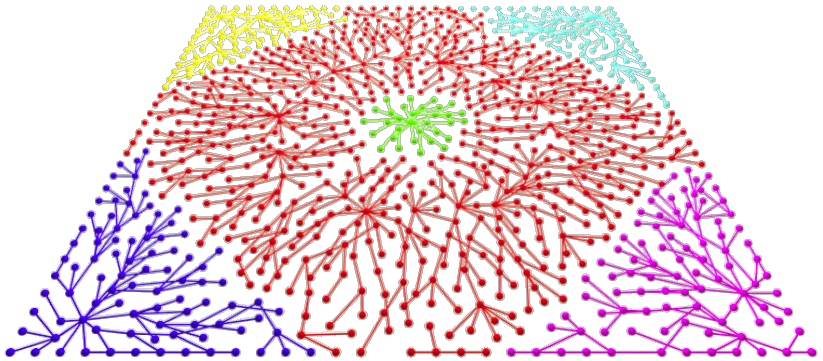
CS233, CME251: Geometric and Topological Data Analysis

Leonidas Guibas
Computer Science Department
Stanford University

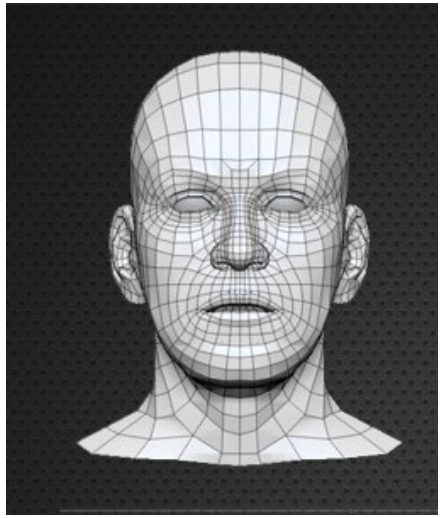




Lecture 18
02 June 2021

Last Time: Shape Differences and Variability

What Exactly is a Shape Difference?



vs.

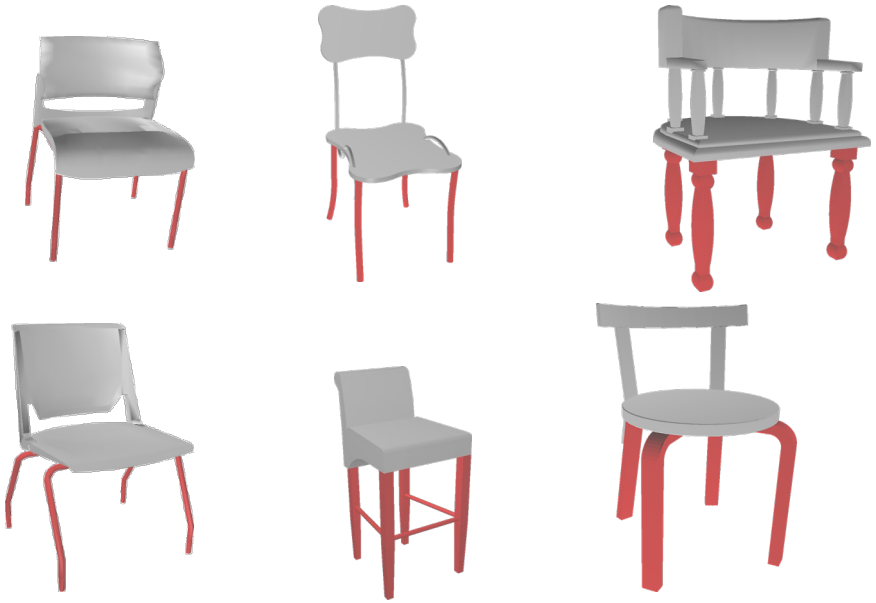
vs.

Making 3D shape differences first class citizens

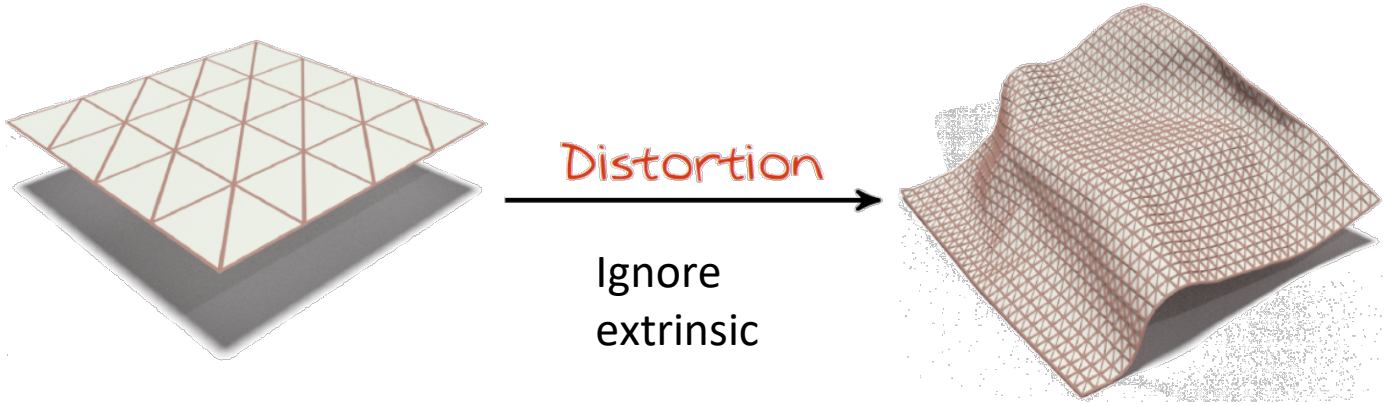
Variation of Structure, Variation of Geometry

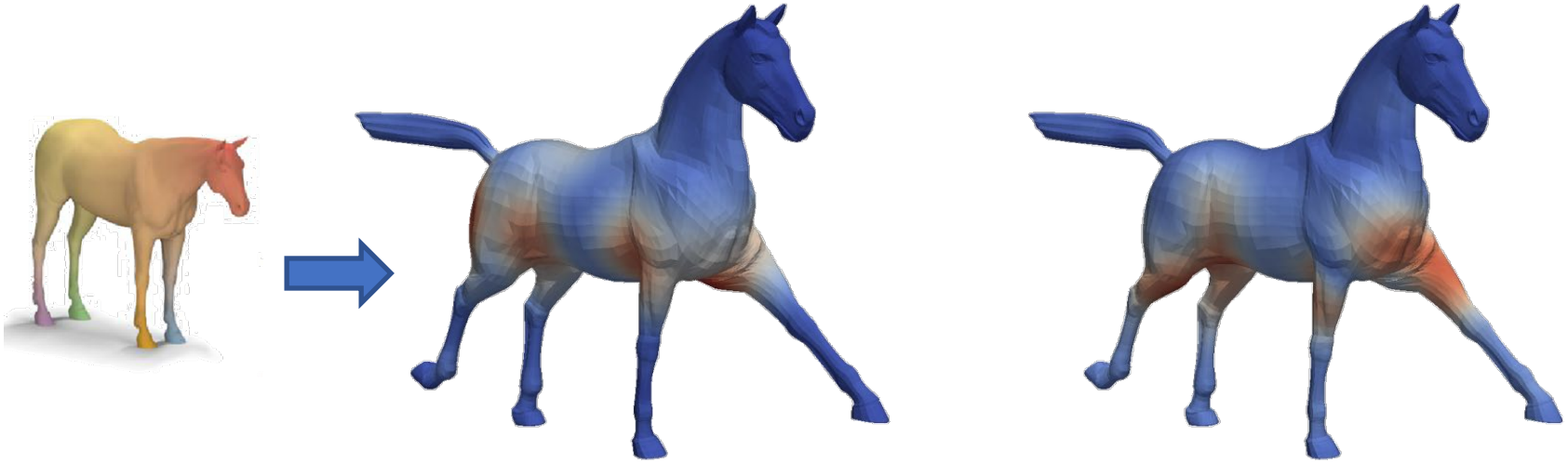
Structure variations

Geometry variations



Intrinsic Changes to a Metric

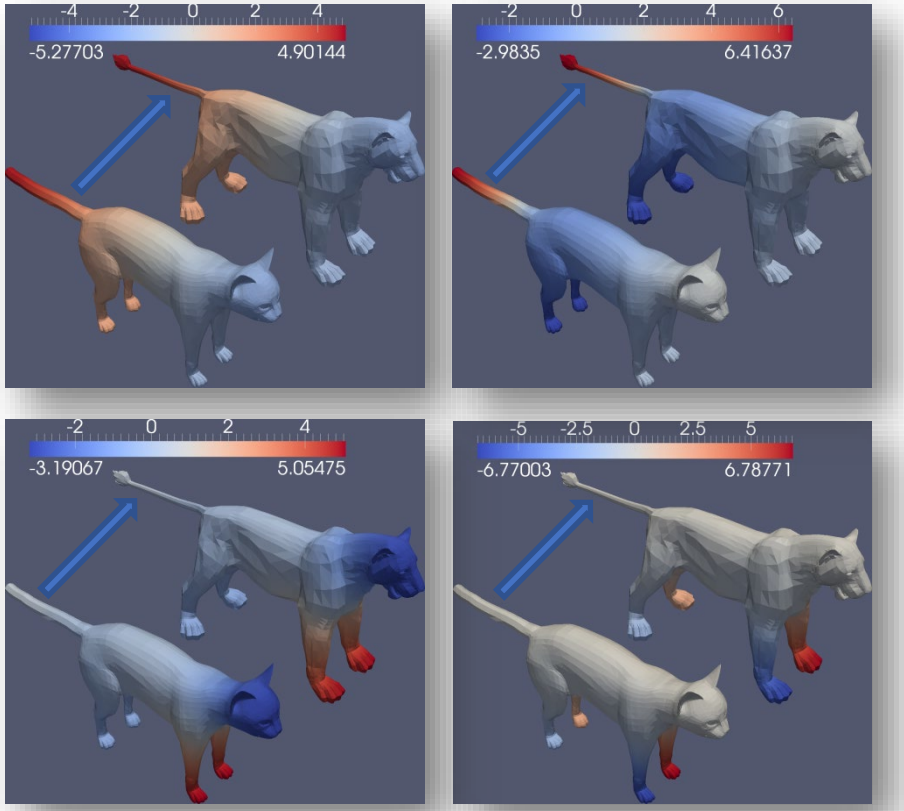




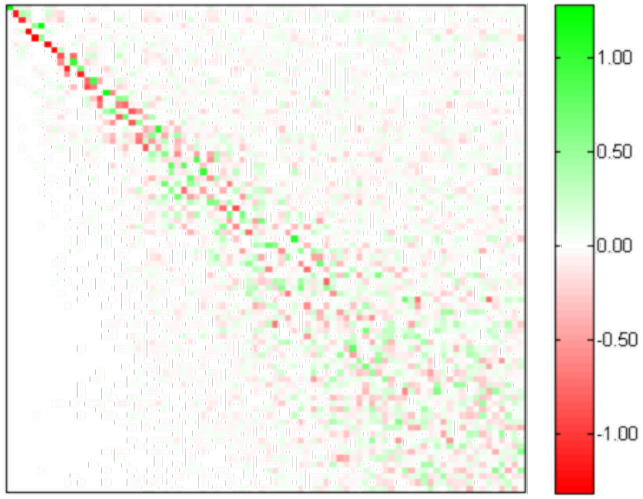
Intrinsic distortions: Area distortion Conformal distortion Length …

Input: Functional Map *F*

from cat to lion



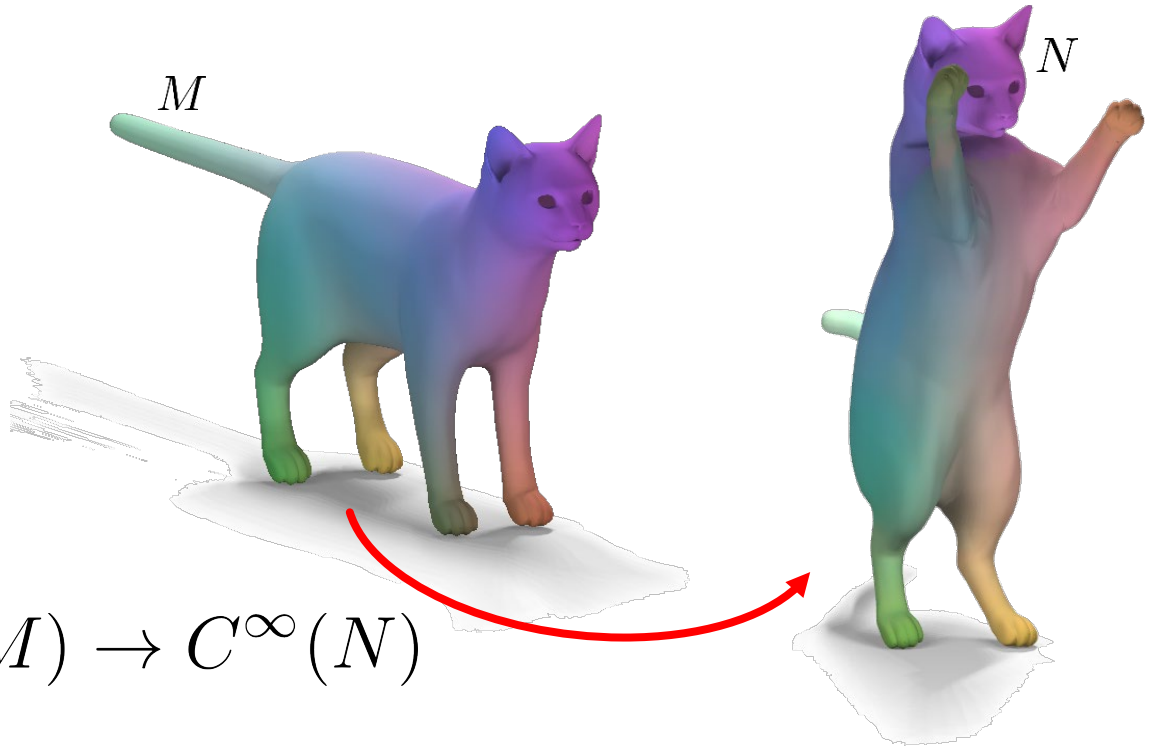
Functions on cat are transferred to lion using *F*



***F* is a linear operator (matrix)**

$$F : L^2(cat) \rightarrow L^2(lion)$$

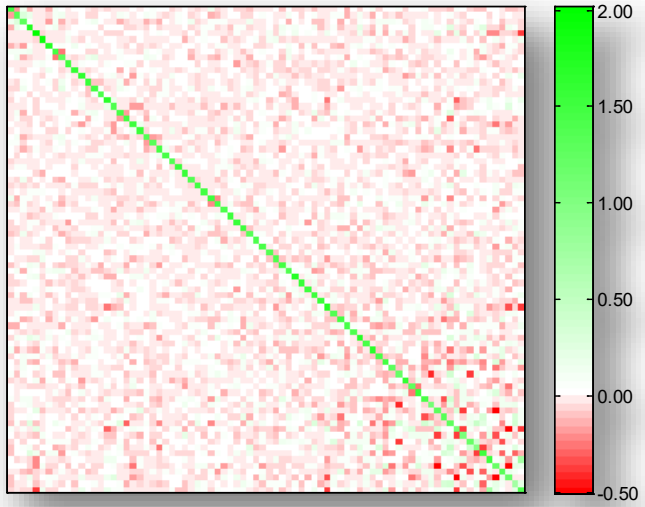
Riesz Representation Theorem



$$\exists V \text{ s.t.}$$
$$\langle F(f), F(g) \rangle^N = \langle f, V(g) \rangle^M \ \forall f, g$$

Shape Difference Operators

V – area-based shape difference

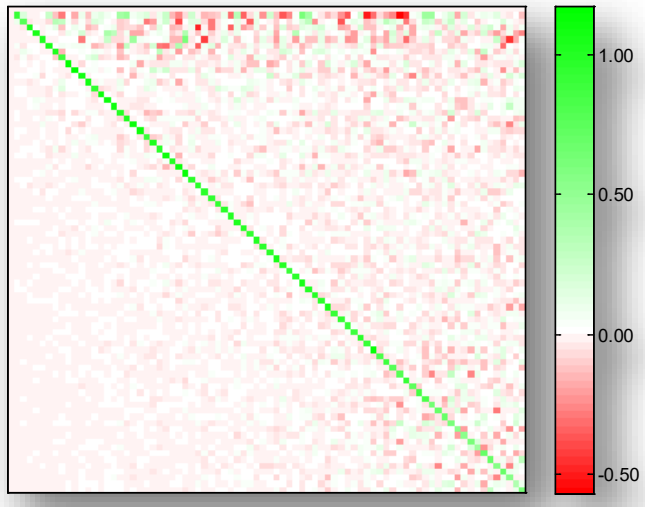


linear operator (matrix)

$$V : L^2(cat) \rightarrow L^2(cat)$$

$$\int_N F(f)F(g) = \int_M fV(g)$$

R – conformal shape difference

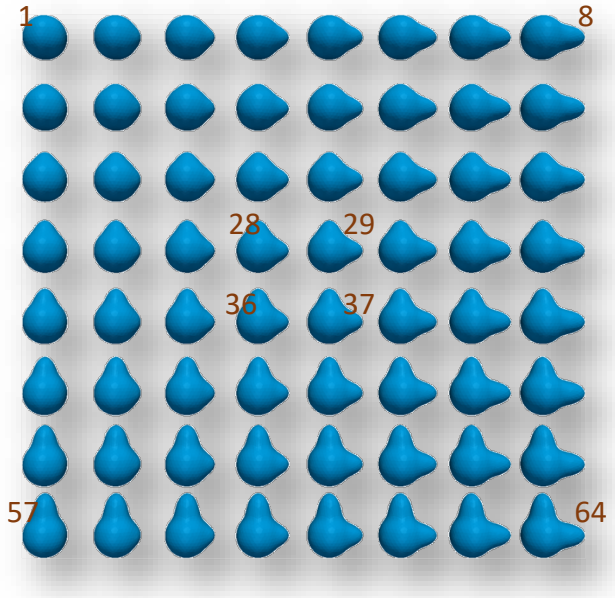


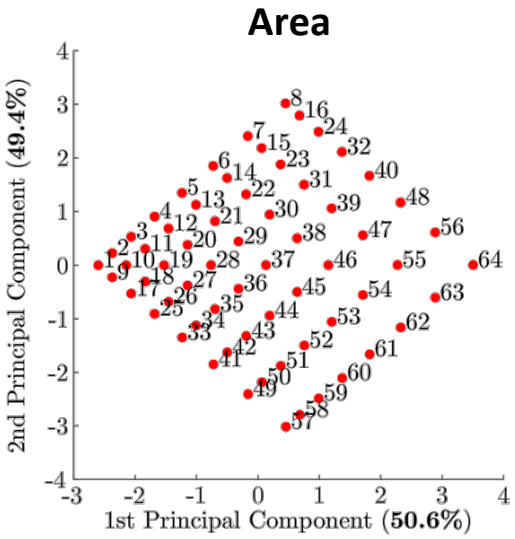
linear operator (matrix)

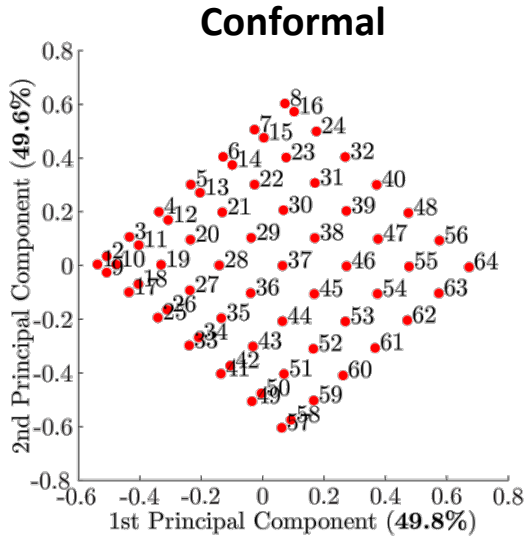
$$R : L^2(cat) \rightarrow L^2(cat)$$

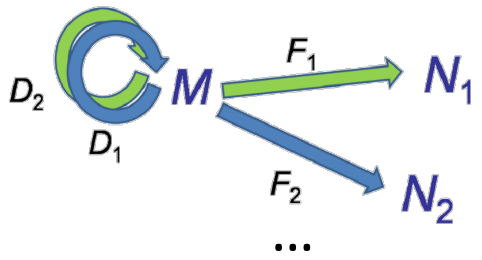
$$\int_N \nabla F(f)\nabla F(g) = \int_M \nabla f \nabla R(g)$$

Intrinsic Shape Space



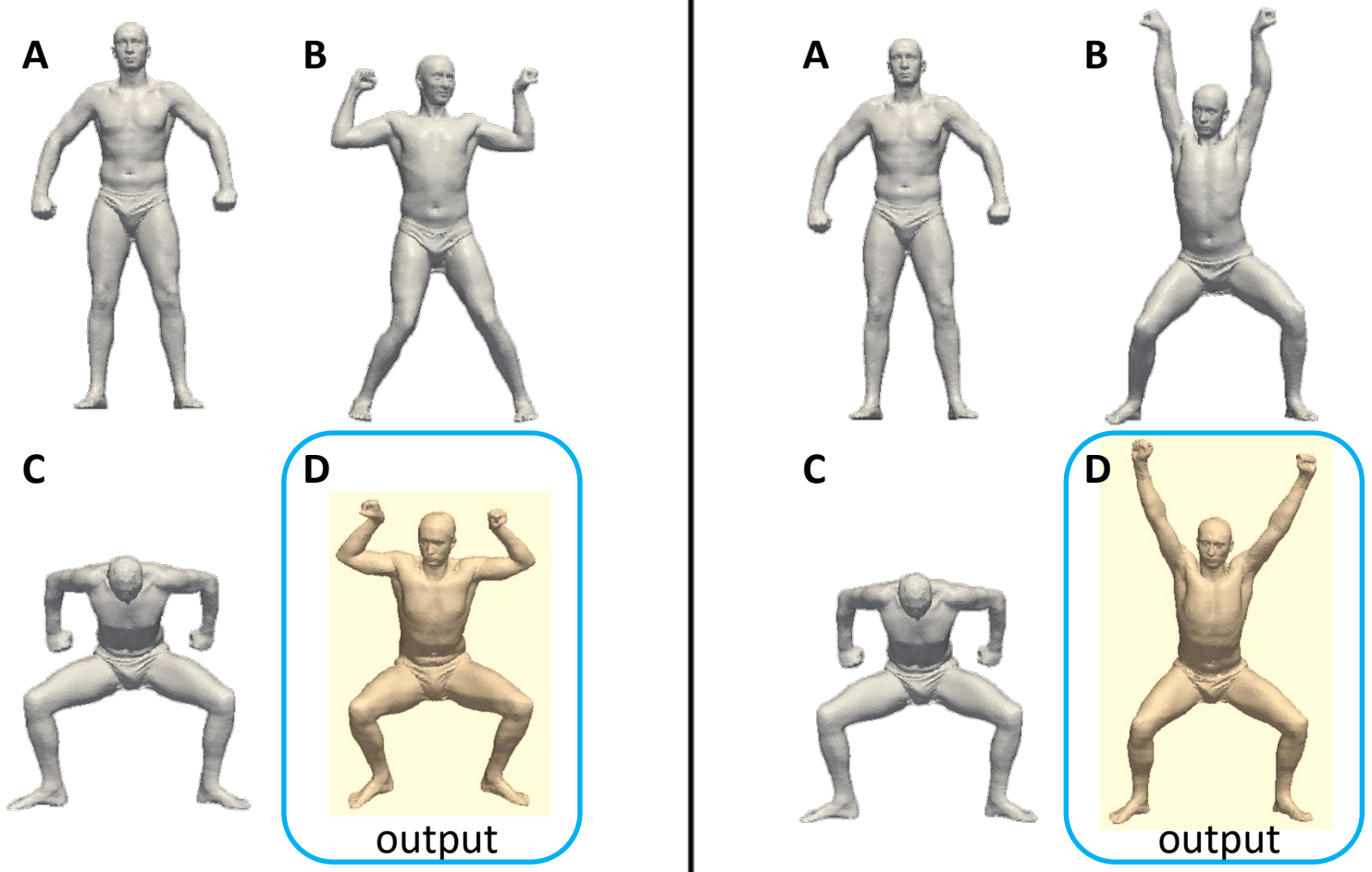






Deformation localization

Shape Analogies

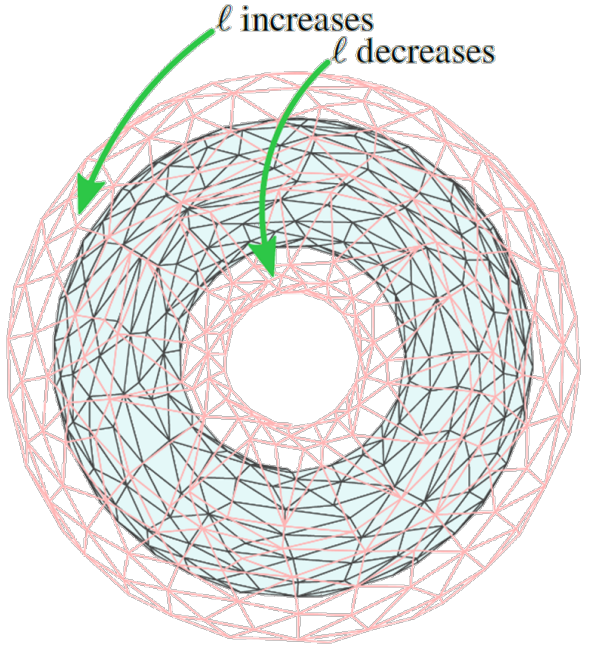


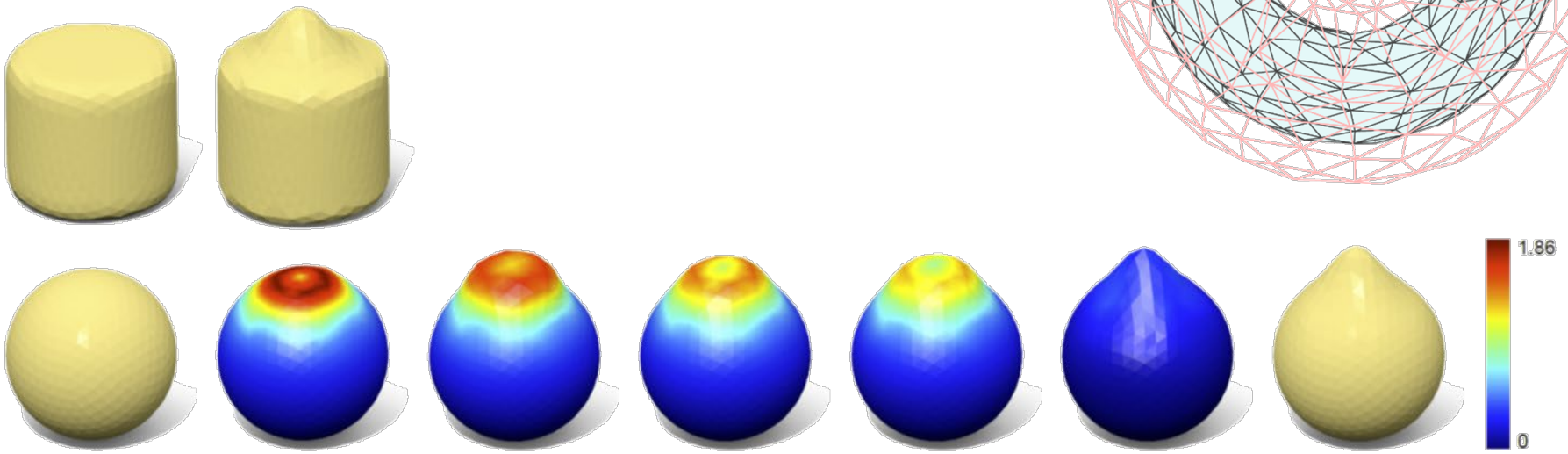
Extrinsic Shape Differences, Version 1

By adding intrinsic differences of an offset surface, we capture extrinsic distortions of the original surface!

Full recovery is provably possible

In practice, challenging optimization problem, especially when the functional basis has been truncated

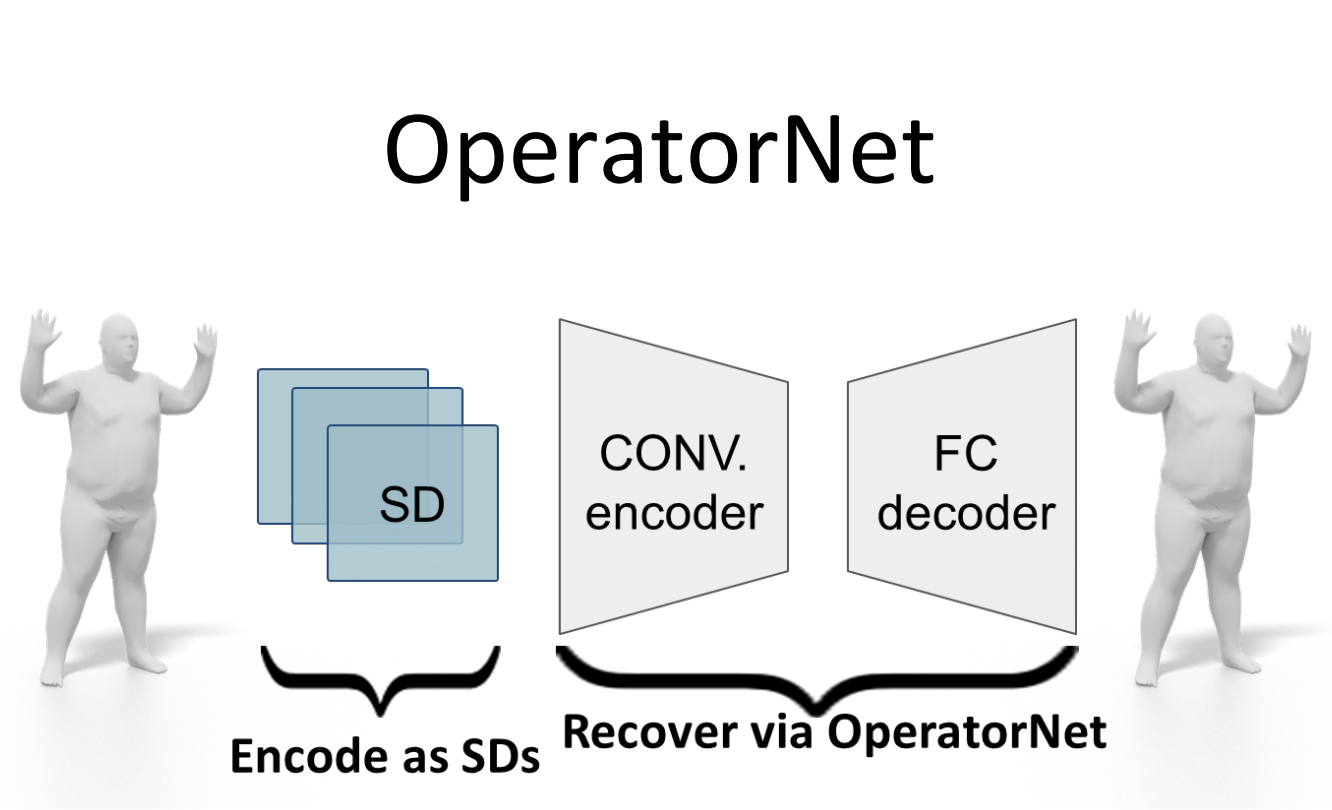




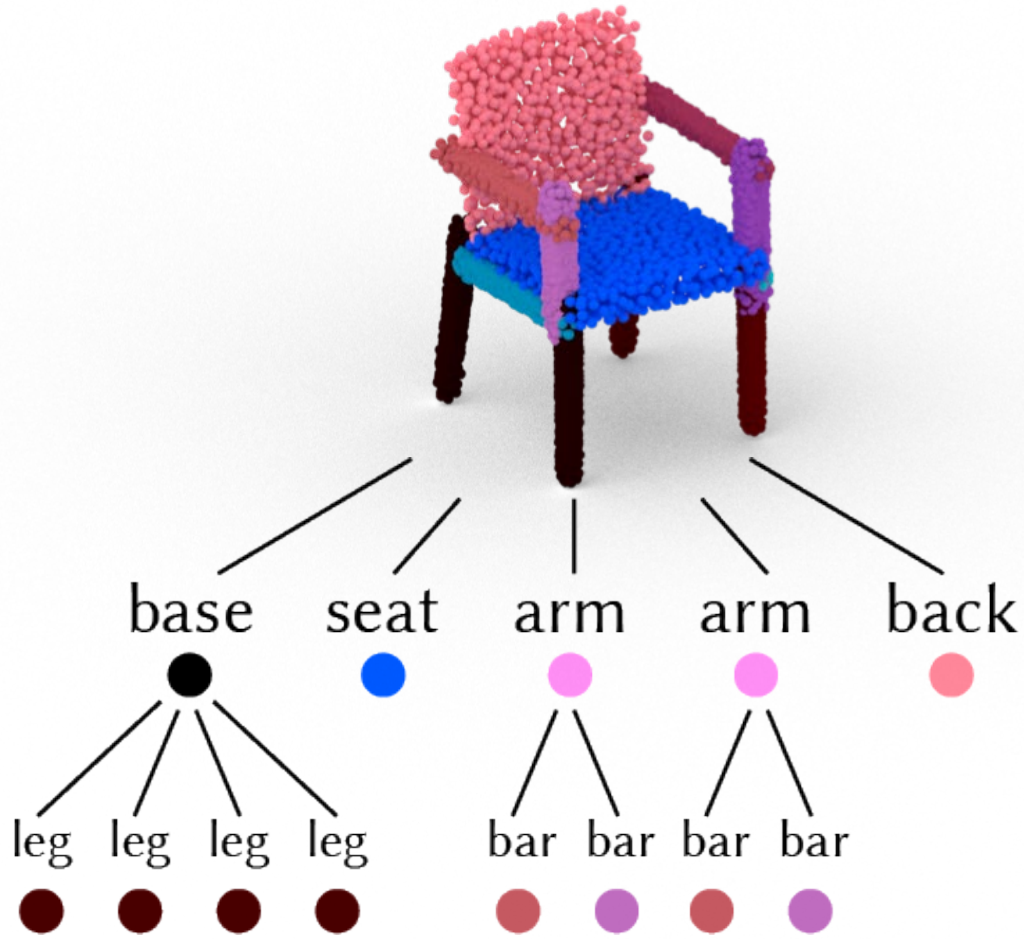
Extrinsic Shape Differences, Version 2

Decode 3D shapes via deep nets directly from shape difference operators

- **Advantages:**
  - **Compact encodings** (small matrices of size)
  - **Natural algebraic** manipulation
  - **Invariant** to rigid transformation
  - Adapted to **convolutional** neural networks
- Applications: shape interpolation, style transfer, up-sampling

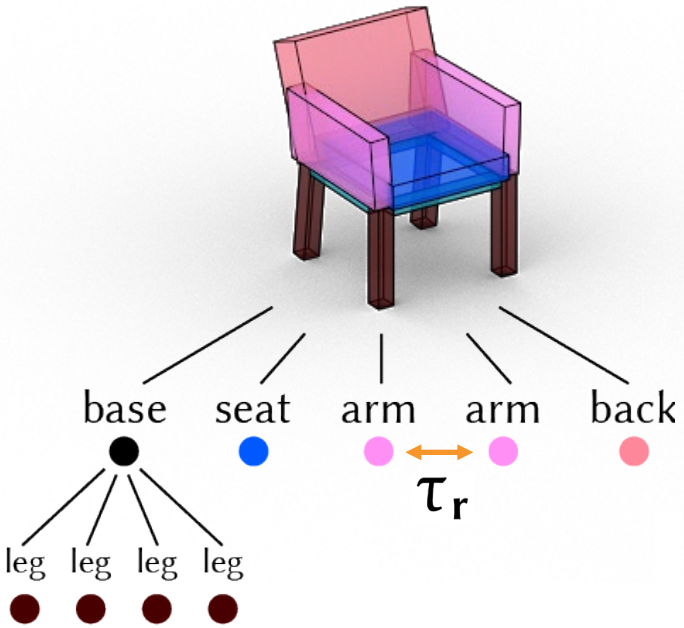


Structure: Part Hierarchy



Architecture Overview: VAE Training

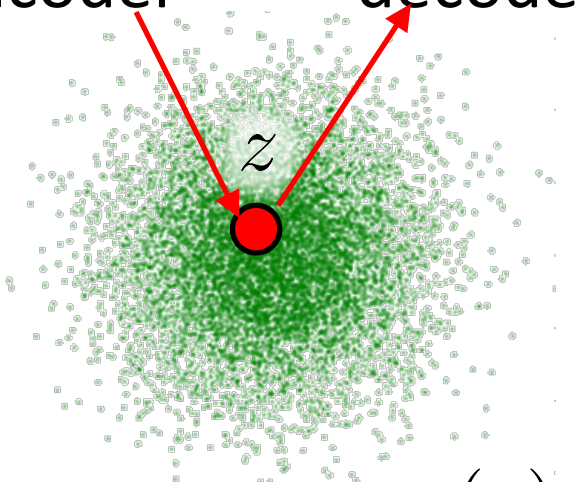
object with structure



reconstruction loss

encoder

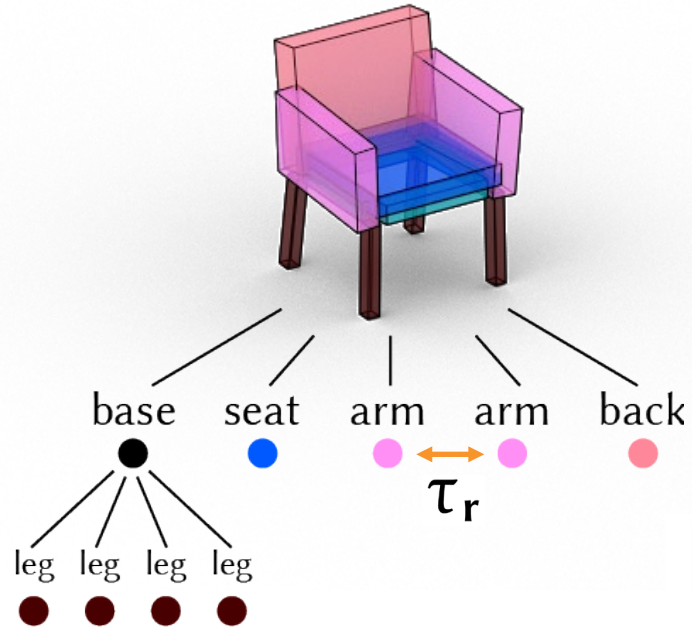
decoder



shape space $p(z)$

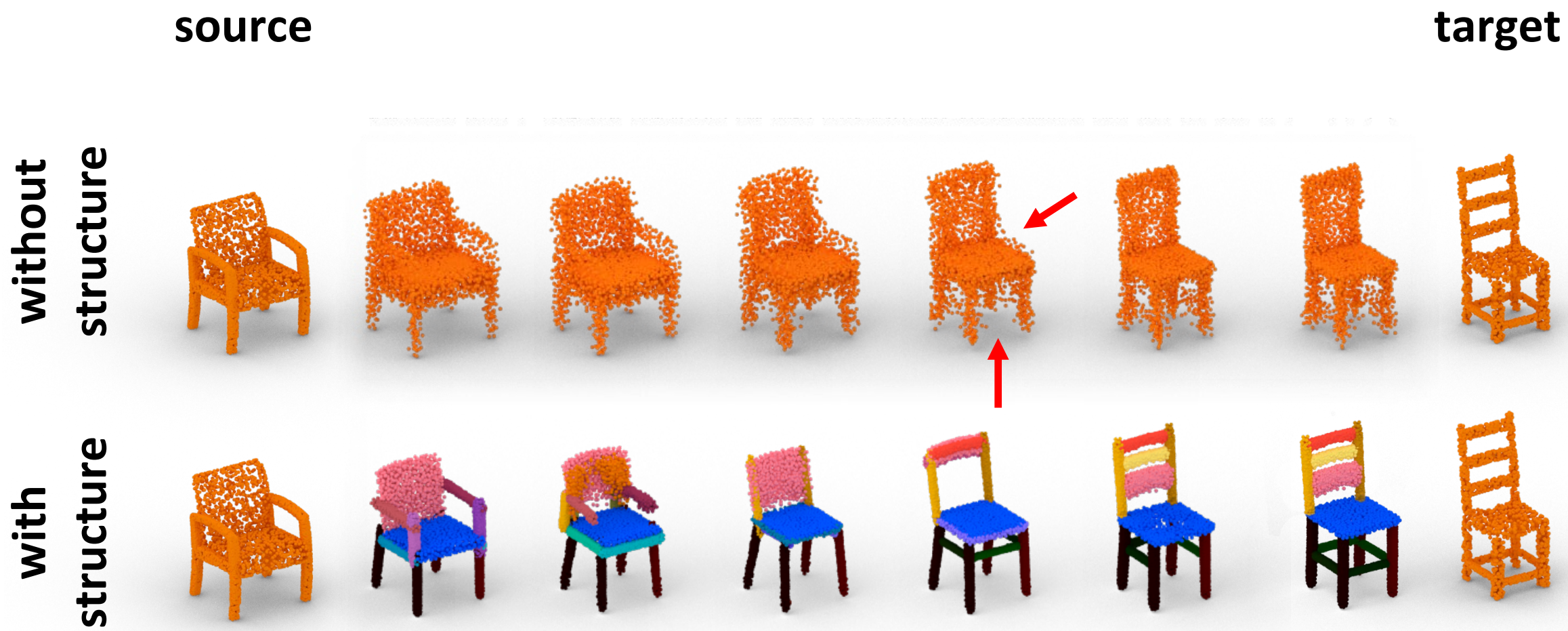
(variational regularization)

Reconstructed object with structure



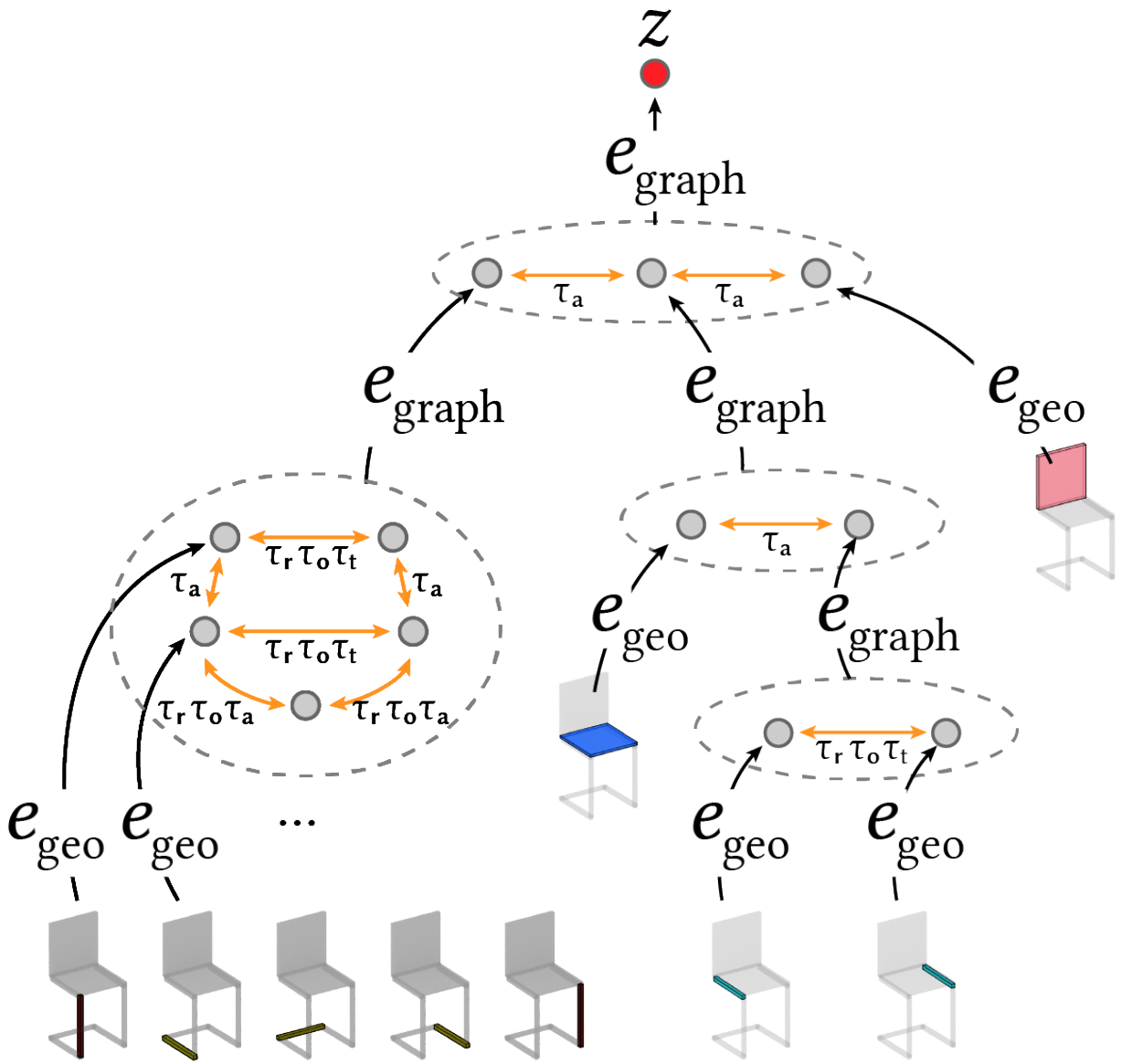
structure consistency loss

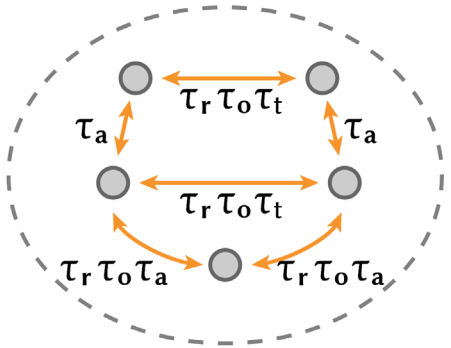
With vs. Without Structure



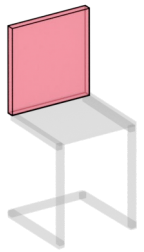
Generation

Hierarchical Graph Encoder



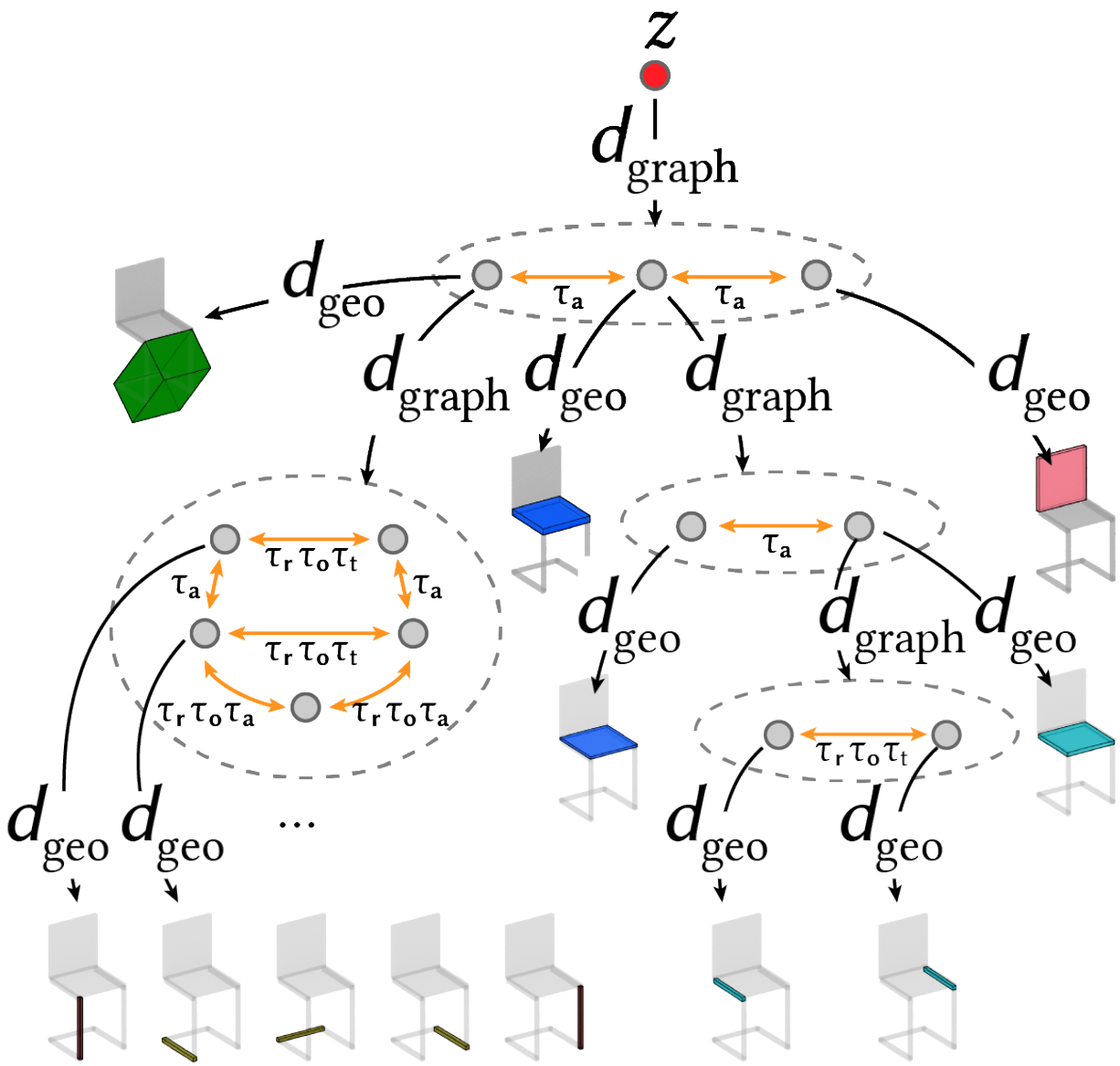


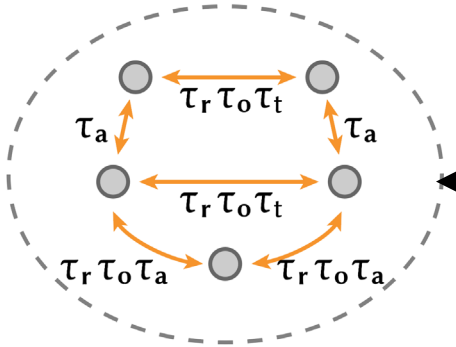
$-e_{graph}\rightarrow$ feature vector



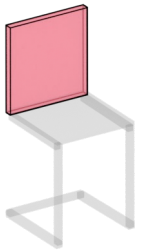
$-e_{geo}\rightarrow$ feature vector

Hierarchical Graph Decoder



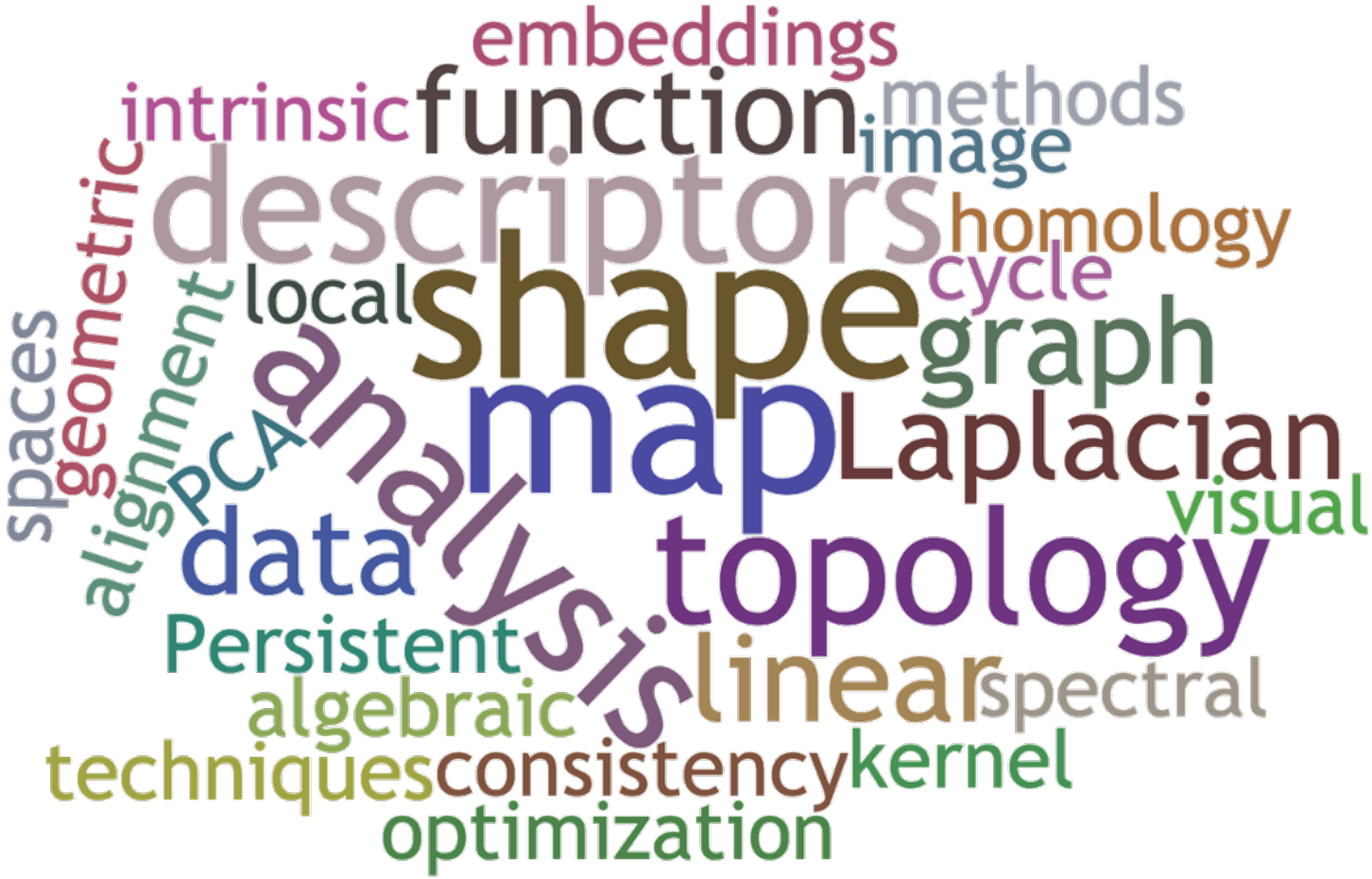


d_{graph} — feature vector



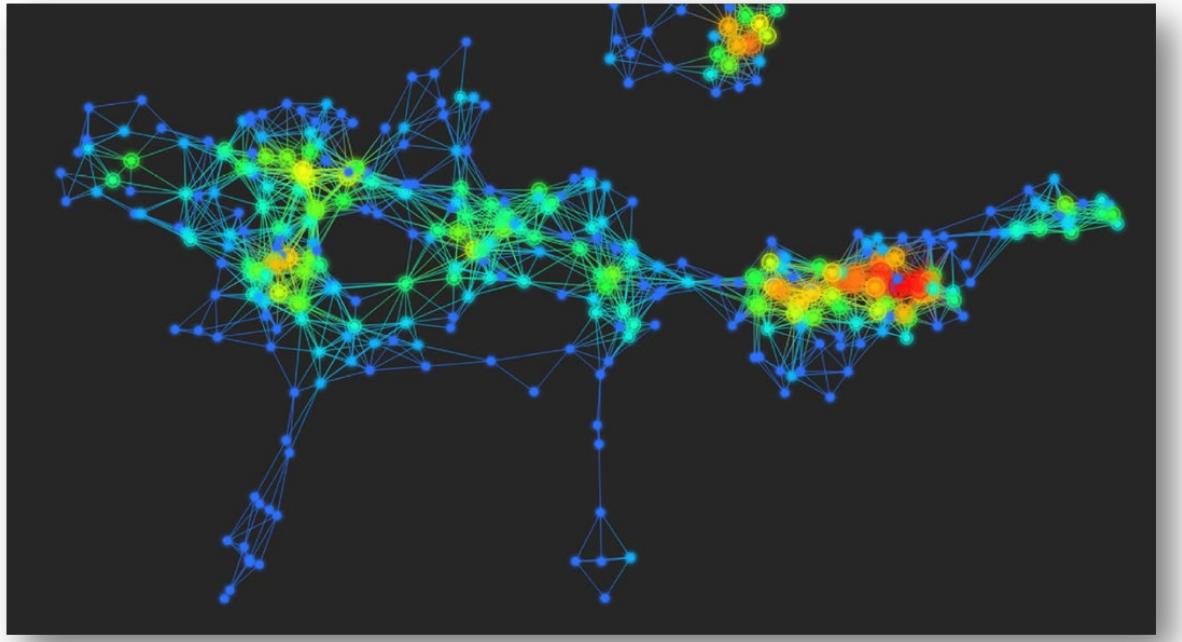
d_{geo} — feature vector

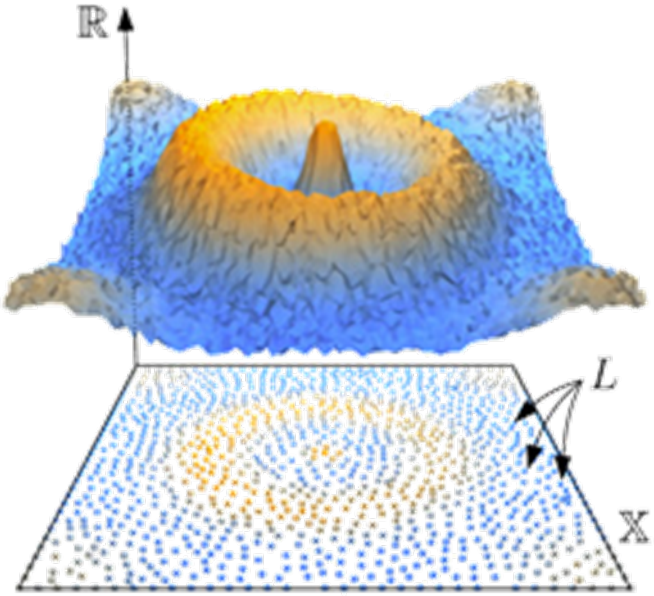
The Course


embeddings
intrinsic
function
methods
image
descriptors
homology
cycle
shape
graph
local
geometric
spaces
alignment
PCA
analysis
map
Laplacian
visual
data
topology
Persistent
linear
spectral
algebraic
techniques
consistency
kernel
optimization

Data Analysis

- (Geometric and Topological Data) (Analysis)
- (Geometric and Topological) (Data Analysis)





Course Survey

CS233 Course Survey

Please submit feedback regarding your experience in CS233: Geometric and Topological Data Analysis this quarter by Thursday, June 3, 2021. We really appreciate your feedback in improving the course for future offerings!

* Required

I am a *

- ○ PhD student
- ○ Master's student
- ○ Undergraduate student

My background/degree is in

- ☐ Computer Science/Electrical Engineering
- ☐ Math/Statistics
- ☐ Other

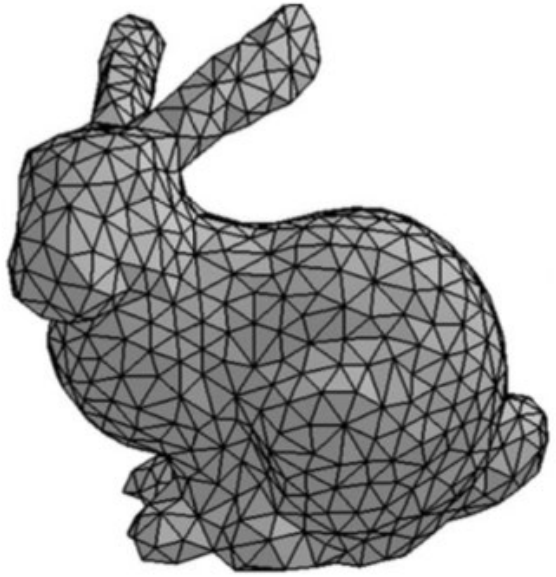
Continuing On …

If you would like to pursue projects or research related to the topics of this class, please get in touch:

guibas@cs.stanford.edu

Today: CNN for Graphs and Meshes

What about Non-Euclidean Domains?

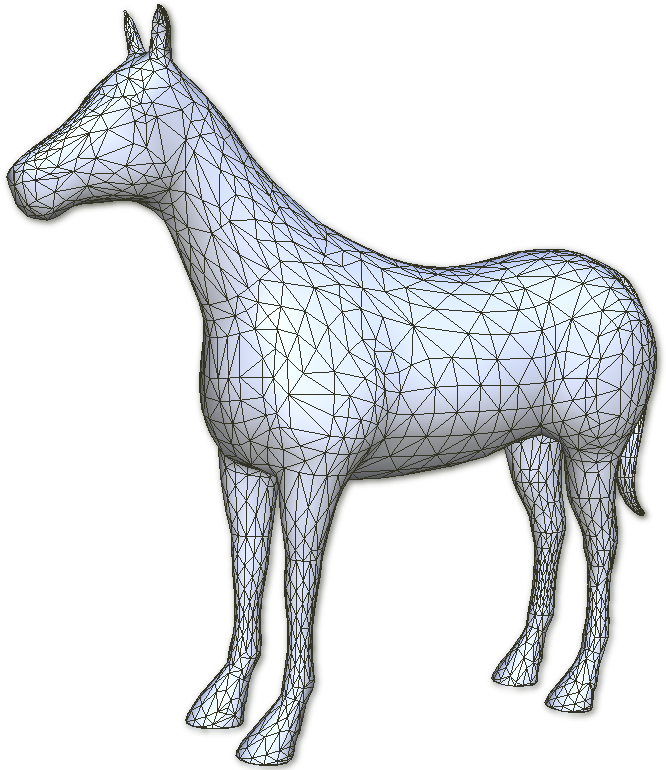


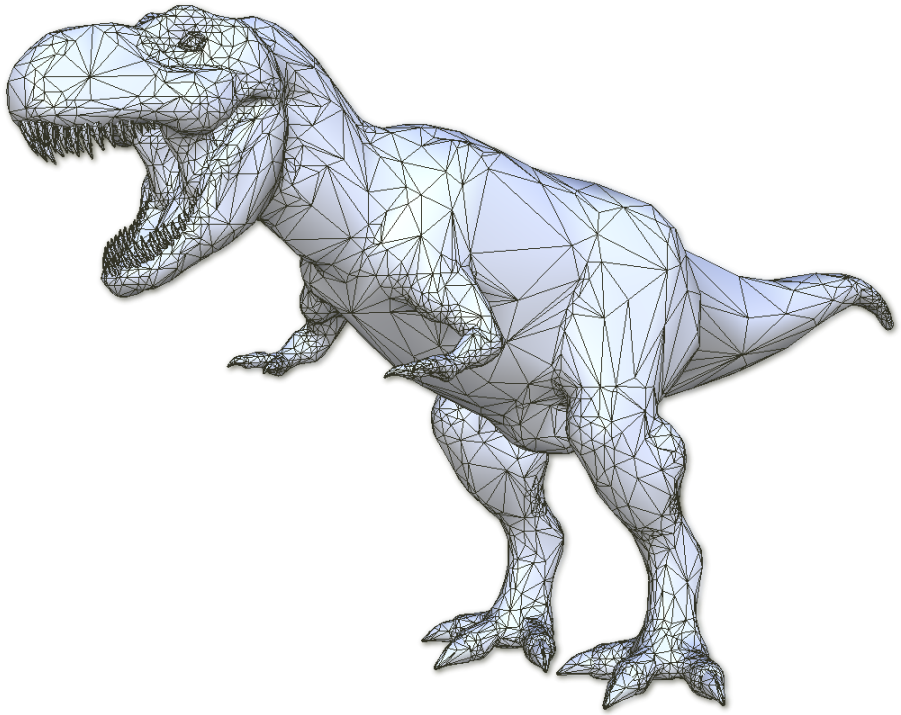
3D shape graph

General graphs

Mesh Representation

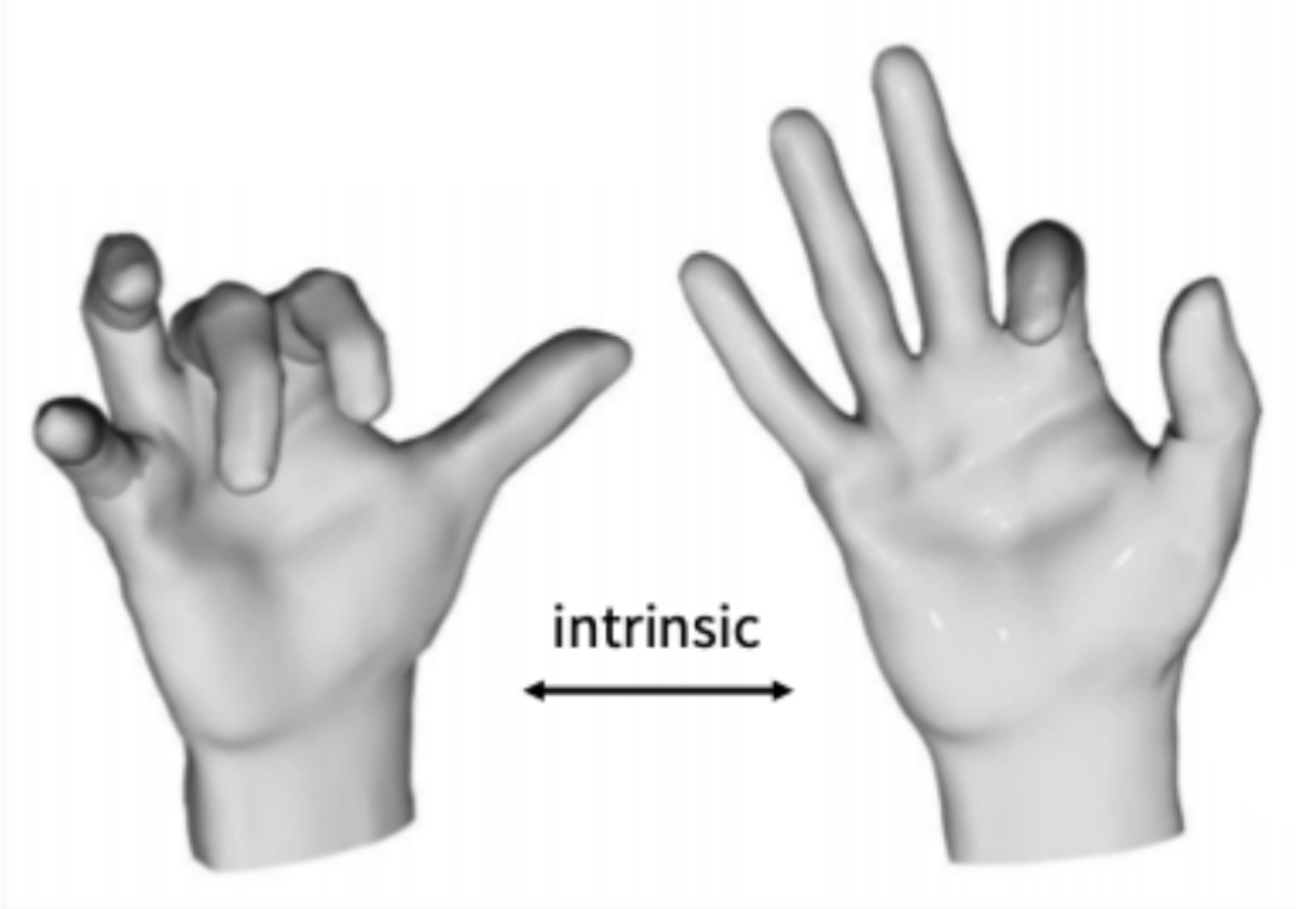
- Effective in encoding fine details





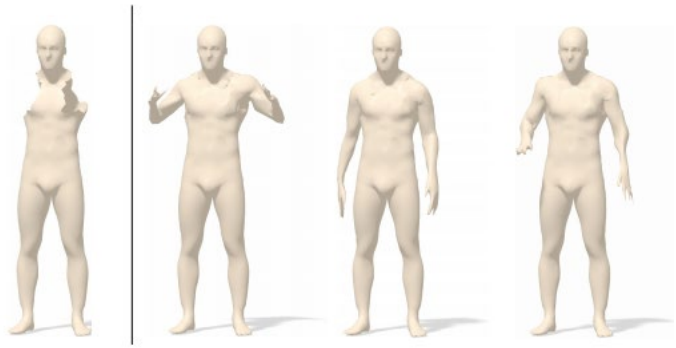
Mesh Representation

- Natural for describing intrinsic similarity

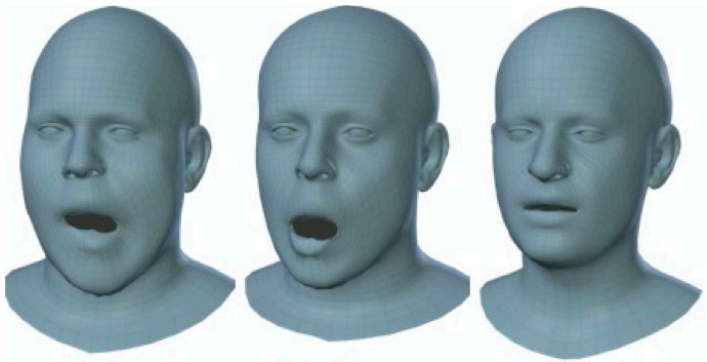


Applications on Shapes

Shape completion

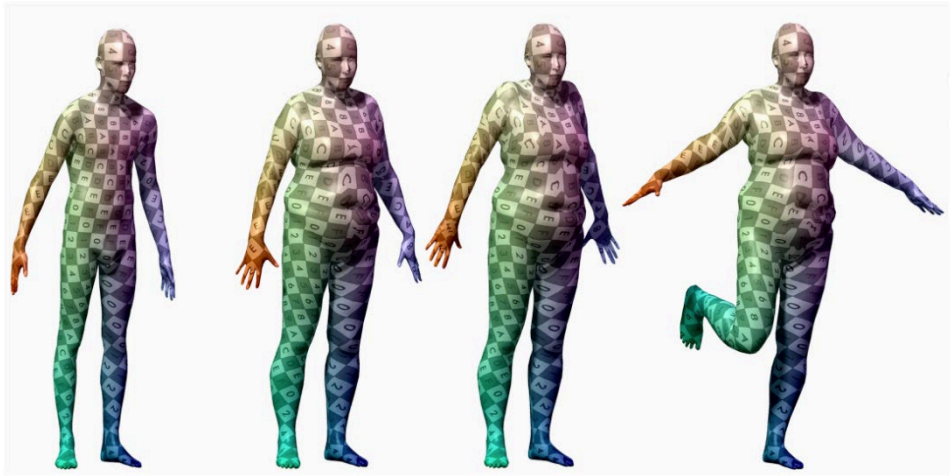


Shape manipulation



Segmentation

Texture transfer

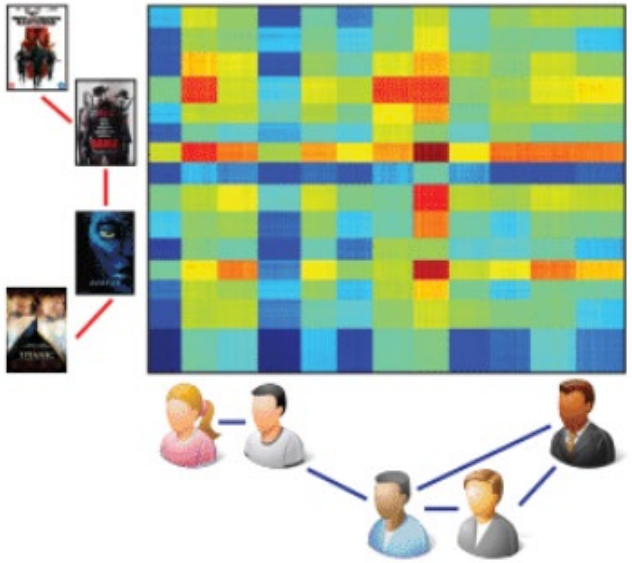


2D —> 3D

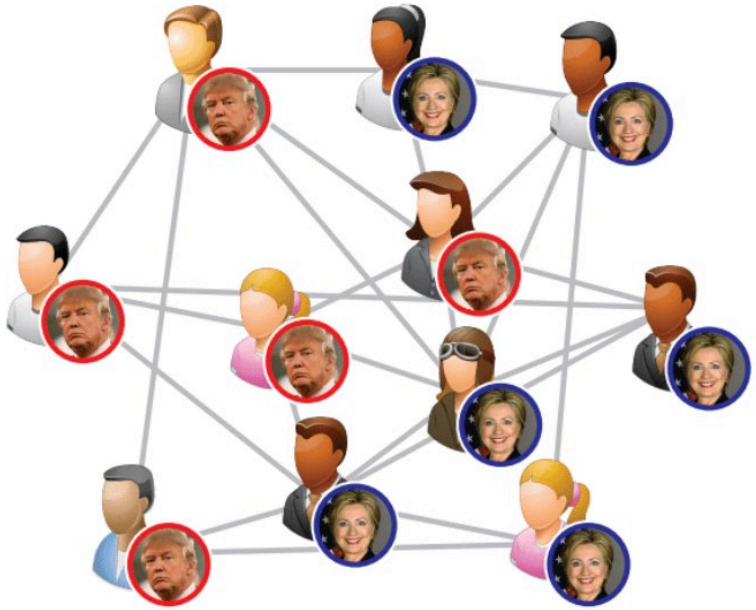
Graph Representation

Graphs

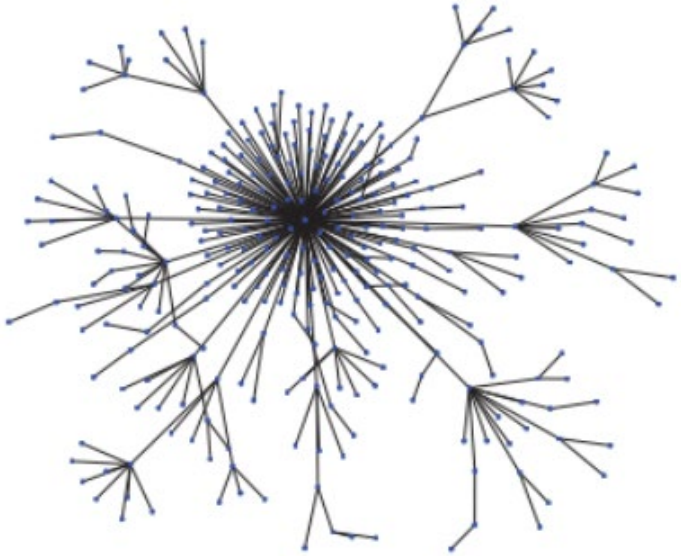
Application on Graphs



Recommendation systems



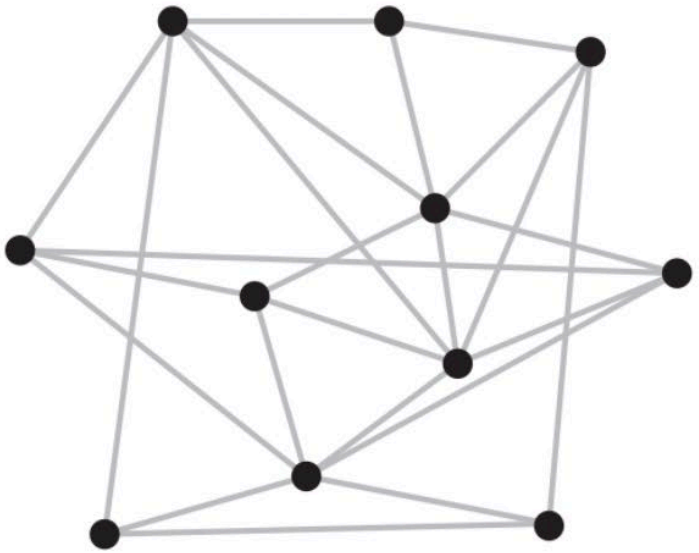
Labeling



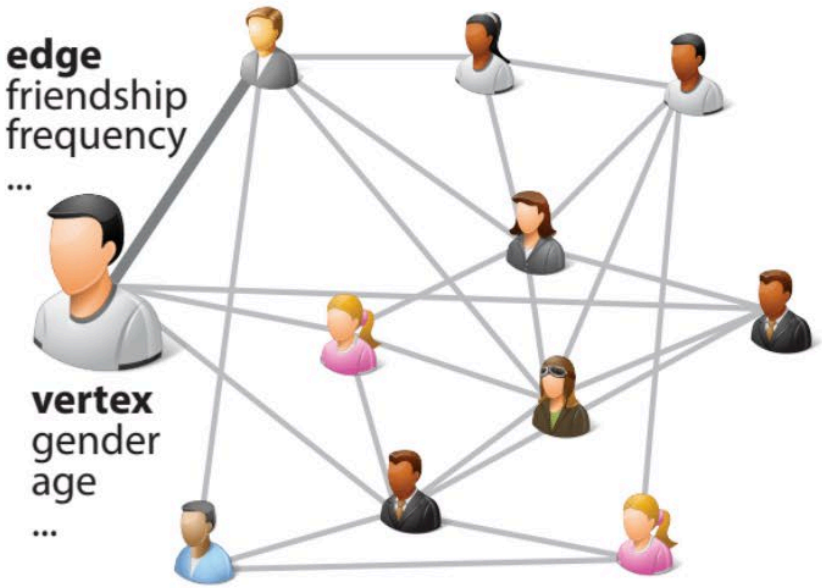
Fake news detection

Challenges

- Domain structure vs. Data on a domain



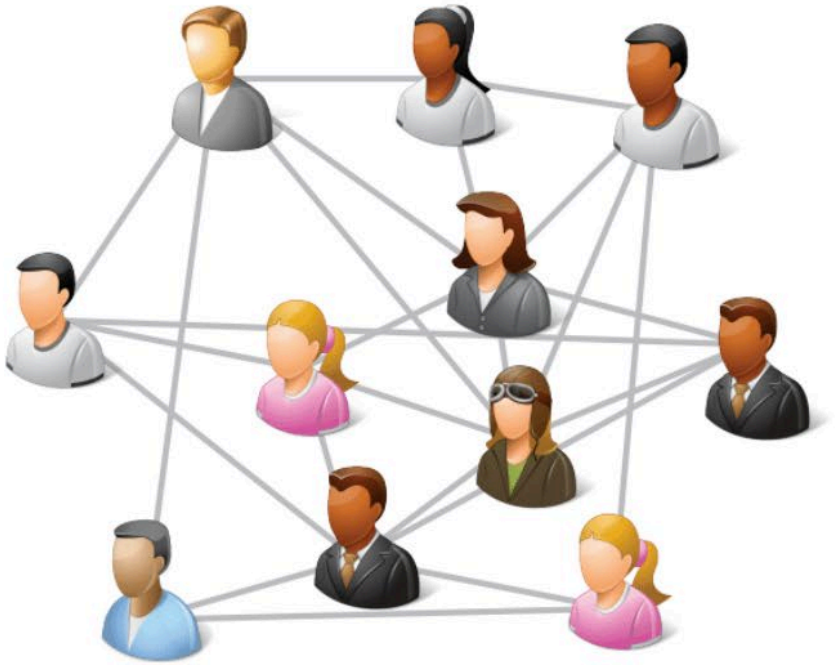
Domain structure



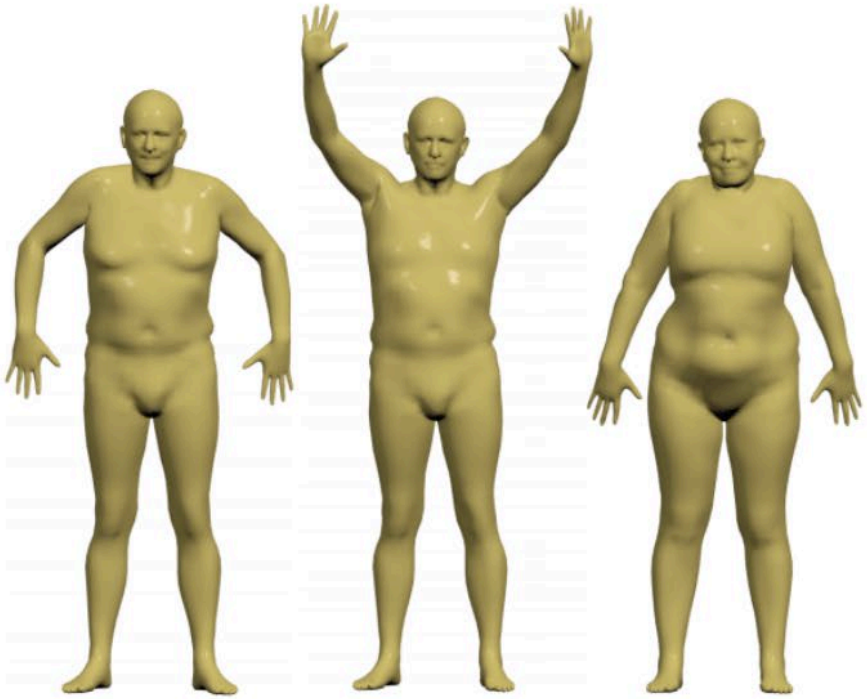
Data on a domain

Challenges

- Fixed vs different domain



Social network
(fixed graph)

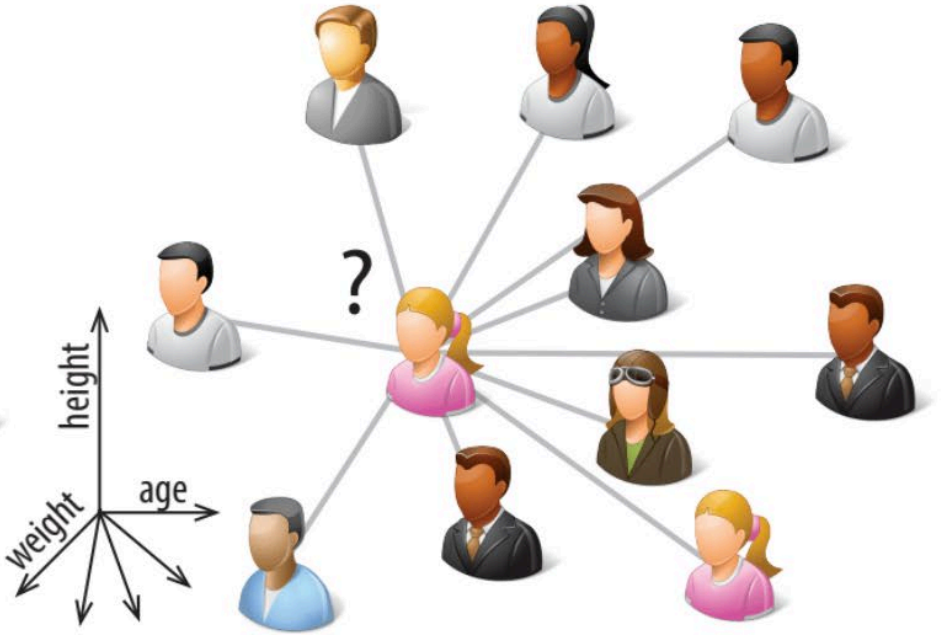


3D shapes
(different manifolds)

Challenges

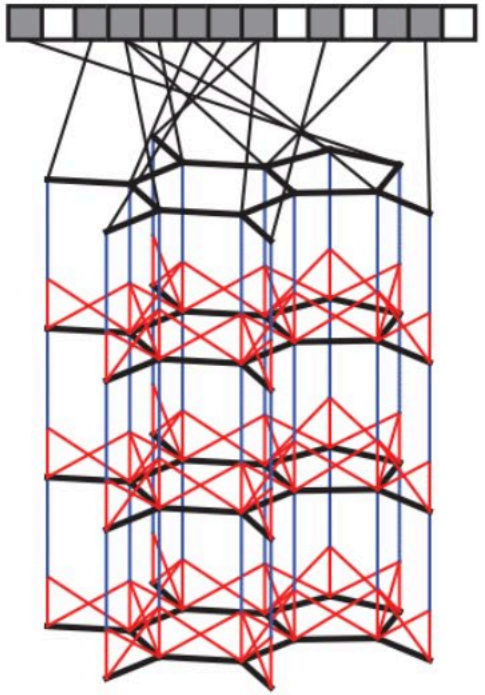
- Known vs unknown graph

Given graph



Learned graph

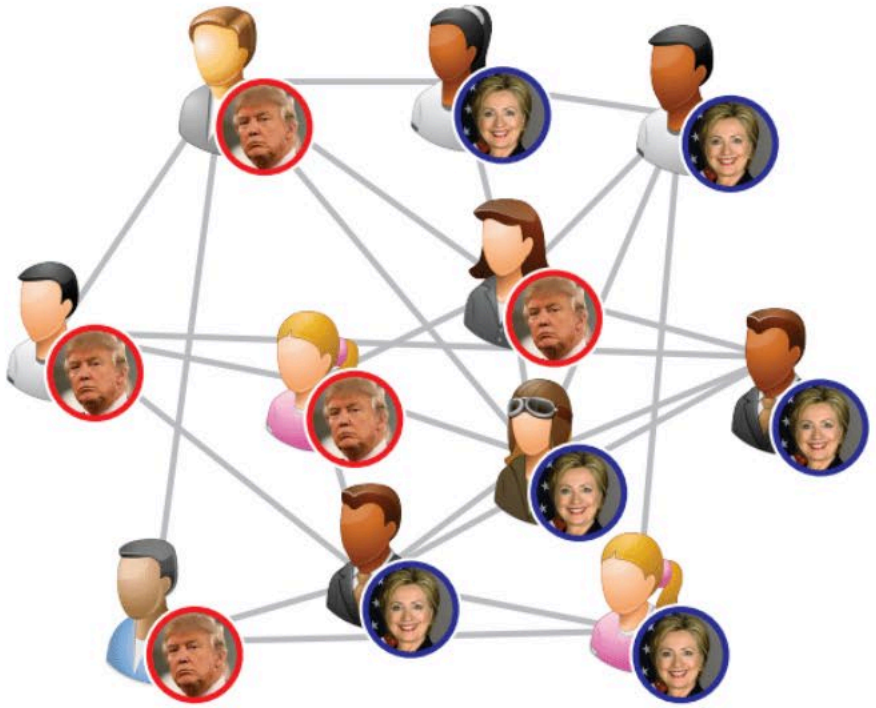
Global Tasks



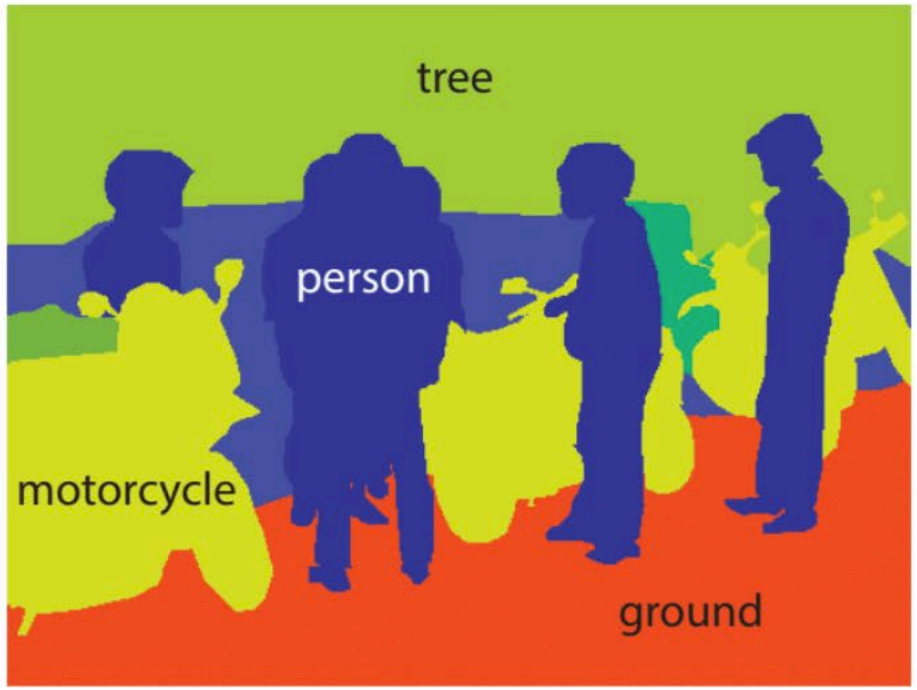
Molecule graph

Image recognition

Vertex-wise Tasks



Social network



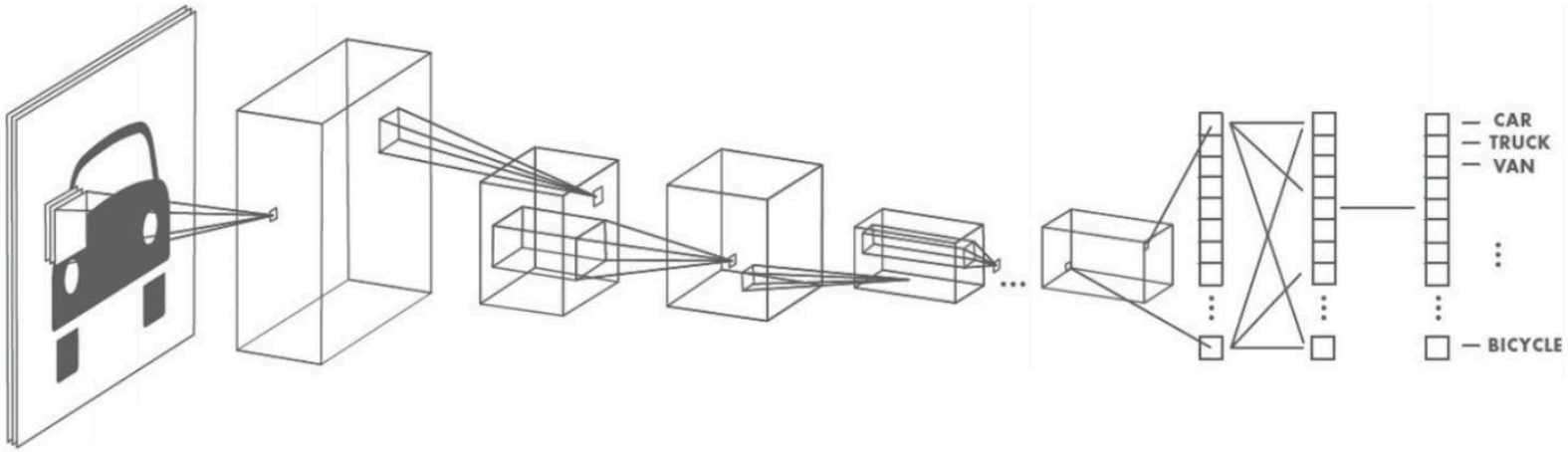
Semantic image segmentation

Different Formulations of non-Euclidean CNNs

Spectral domain

Spatial domain

Key Properties of CNN



- ☺ Convolutional filters **(Translation invariance+Self-similarity)**
- ☺ Multiple layers **(Compositionality)**
- ☺ Filters localized in space **(Locality)**
- ☺ $\mathcal{O}(1)$ parameters per filter (independent of input image size n)
- ☺ $\mathcal{O}(n)$ complexity per layer (filtering done in the spatial domain)
- ☺ $\mathcal{O}(\log n)$ layers in classification tasks

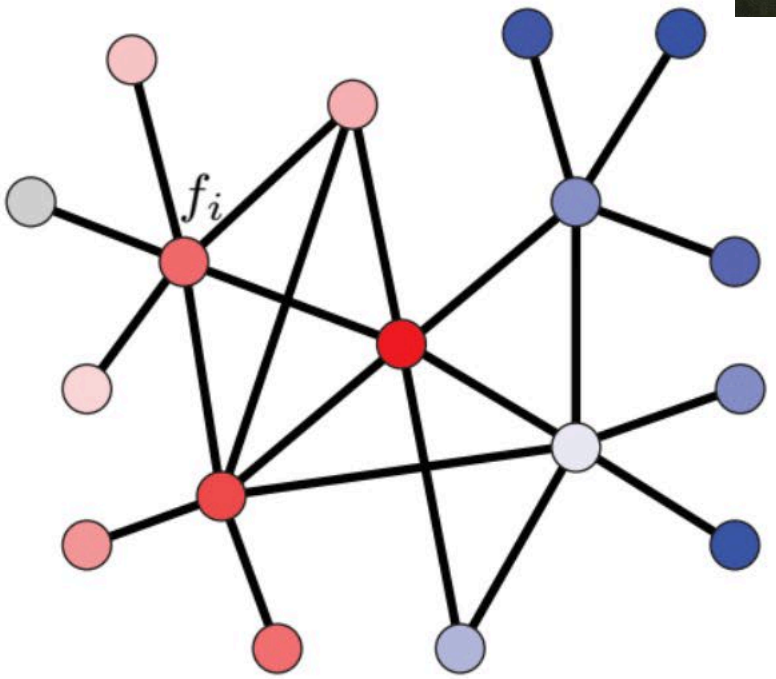
Spectral Methods: Laplacian is Key

- Weighted undirected graph $\mathcal{G}$ with vertices $\mathcal{V} = \{1, \ldots, n\}$, edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ and edge weights $w_{ij} \geq 0$ for $(i, j) \in \mathcal{E}$

- Functions over the vertices $L^2(\mathcal{V}) = \{f : \mathcal{V} \to \mathbb{R}\}$ represented as vectors $\mathbf{f} = (f_1, \ldots, f_n)$

- Hilbert space with inner product

$$\langle f, g \rangle_{L^2(\mathcal{V})} = \sum_{i \in \mathcal{V}} f_i \, g_i = \mathbf{f}^\top \mathbf{g}$$

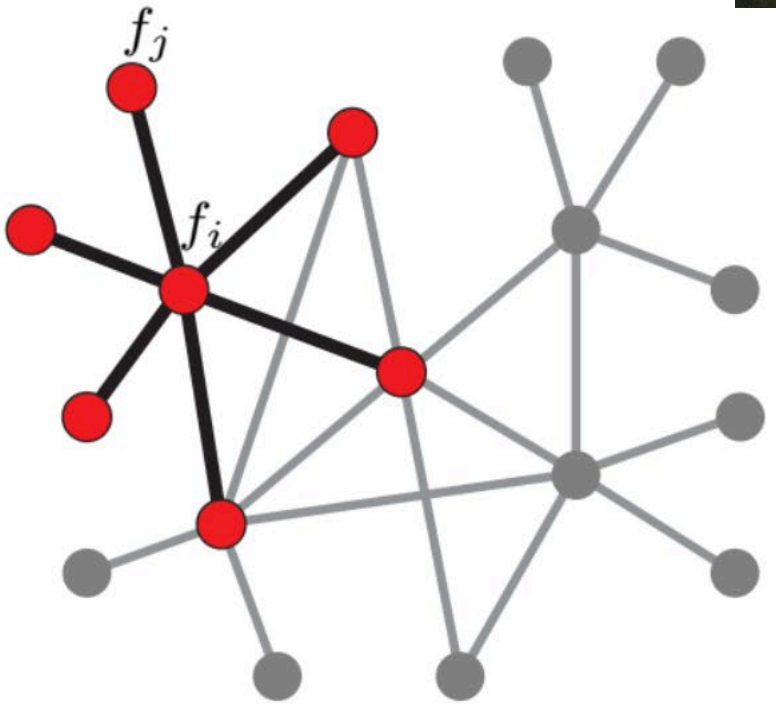


Spectral Methods: Laplacian is Key

- Unnormalized Laplacian $\Delta : L^2(\mathcal{V}) \to L^2(\mathcal{V})$

$$(\Delta f)_i = \sum_{j:(i,j)\in\mathcal{E}} w_{ij}(f_i - f_j)$$

(up to scale) difference between f and its local average



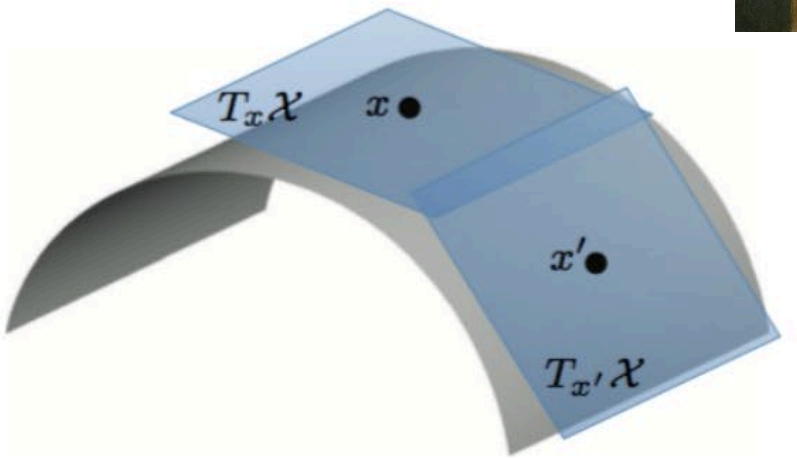
- Represented as a positive semi-definite $n \times n$ matrix $\mathbf{\Delta} = \mathbf{D} - \mathbf{W}$ where $\mathbf{W} = (w_{ij})$ and $\mathbf{D} = \mathrm{diag}(\sum_{j\neq i} w_{ij})$

Spectral Methods: Laplacian is Key

- Manifold $\mathcal{X}$ = topological space
- No global Euclidean structure
- Tangent plane $T_x\mathcal{X}$ = local Euclidean representation of manifold $\mathcal{X}$ around x
- Riemannian metric

$$\langle \cdot, \cdot \rangle_{T_x\mathcal{X}} : T_x\mathcal{X} \times T_x\mathcal{X} \to \mathbb{R}$$

depending smoothly on x

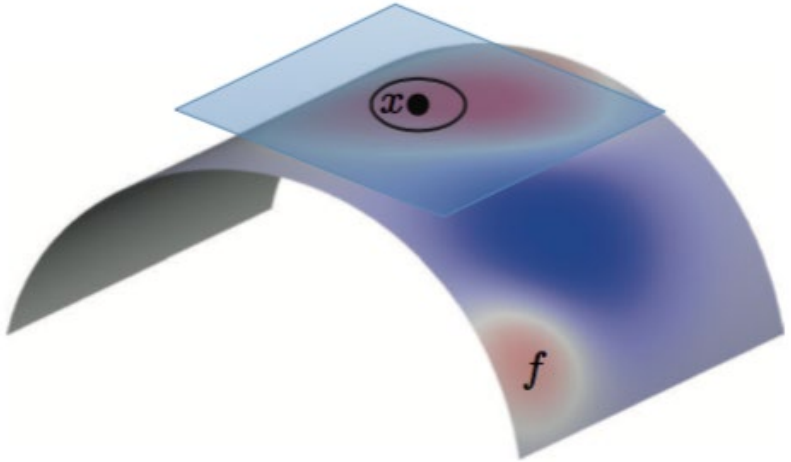


Spectral Methods: Laplacian is Key

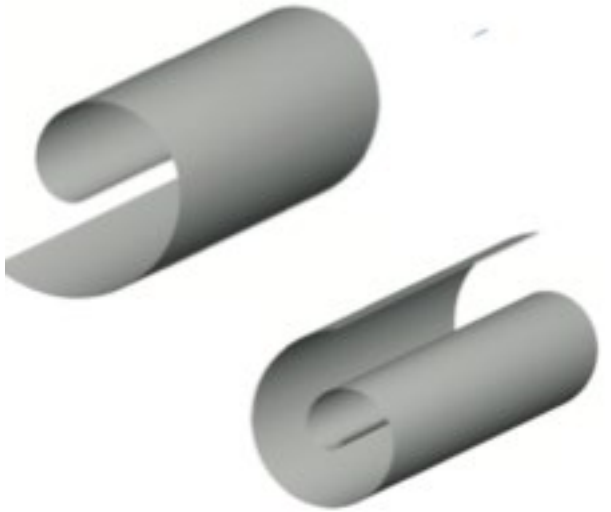
- Laplacian $\Delta : L^2(\mathcal{X}) \to L^2(\mathcal{X})$

$$\Delta f = -\mathrm{div}(\nabla f)$$

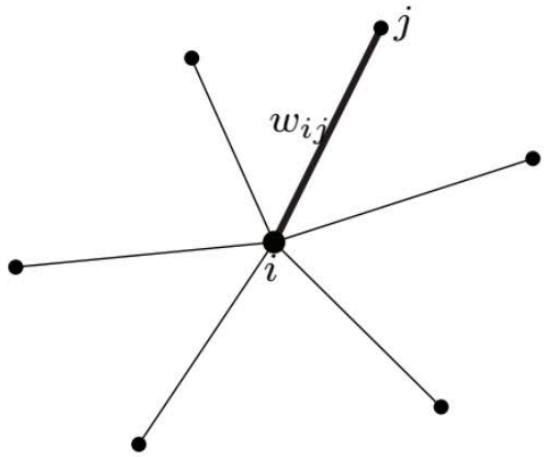
"difference between $f(x)$ and average value of f around x"



- Intrinsic (expressed solely in terms of the Riemannian metric)
- Isometry-invariant
- Positive semidefinite

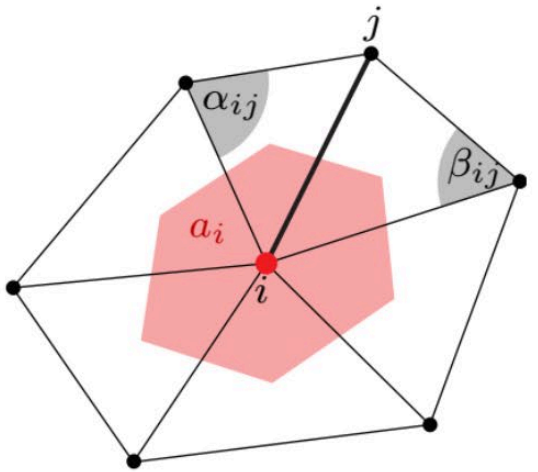


Going non-Euclidean: Laplacian is Key



Undirected graph $(\mathcal{V}, \mathcal{E})$

$$(\Delta f)_i \approx \sum_{(i,j)\in\mathcal{E}} w_{ij}(f_i - f_j)$$



Triangular mesh $(\mathcal{V}, \mathcal{E}, \mathcal{F})$

$$(\Delta f)_i \approx \frac{1}{a_i} \sum_{(i,j)\in\mathcal{E}} \frac{\cot \alpha_{ij} + \cot \beta_{ij}}{2} (f_i - f_j$$

a_i = local area element

In matrix-vector notation

$$\mathbf{\Delta f} = \mathbf{A}^{-1}(\mathbf{D} - \mathbf{W})\mathbf{f}$$

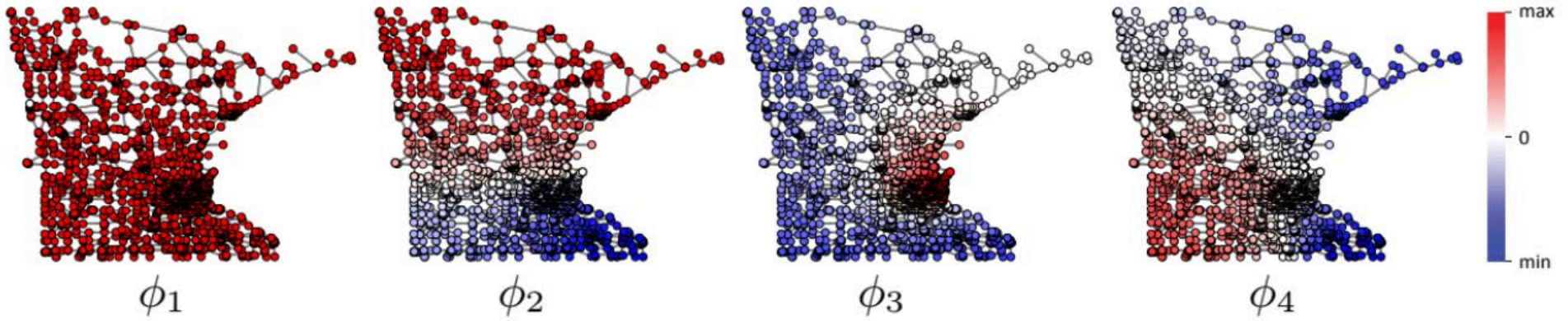
where $\mathbf{f} = (f_1, \ldots, f_n)^\top$, $\mathbf{W}$ is the stiffness matrix, $\mathbf{A} = \mathrm{diag}(a_1, \ldots, a_n)$ is the mass matrix, and $\mathbf{D} = \mathrm{diag}(\sum_{j\neq 1} w_{1j}, \ldots, \sum_{j\neq n} w_{nj})$

Going non-Euclidean: Laplacian is Key

Eigendecomposition of a graph Laplacian

$$\mathbf{\Delta} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top$$

where $\mathbf{\Phi} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n)$ are orthogonal eigenvectors ($\mathbf{\Phi}^\top\mathbf{\Phi} = \mathbf{I}$) and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ the corresponding non-negative eigenvalues



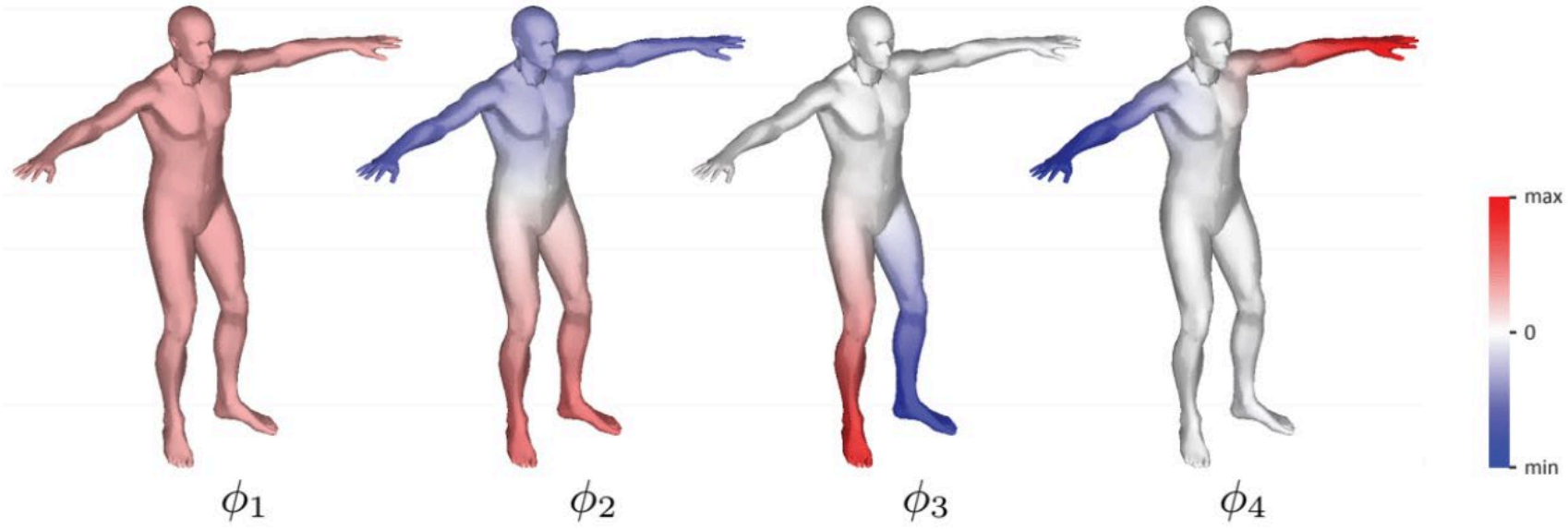
First eigenfunctions of a graph Laplacian

Going non-Euclidean: Laplacian is Key

Eigendecomposition of a graph Laplacian

$$\boldsymbol{\Delta} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^\top$$

where $\boldsymbol{\Phi} = (\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n)$ are orthogonal eigenvectors ($\boldsymbol{\Phi}^\top\boldsymbol{\Phi} = \mathbf{I}$) and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ the corresponding non-negative eigenvalues

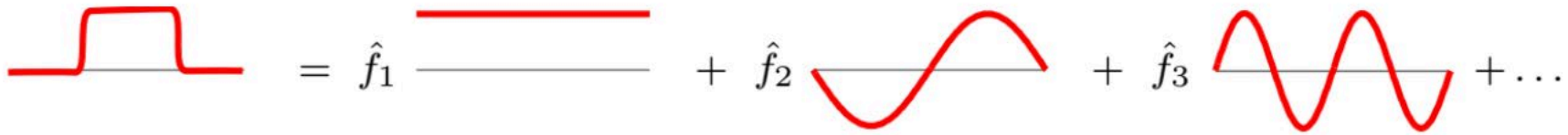


First eigenfunctions of a manifold Laplacian

Fourier Analysis on Euclidean Spaces

A function $f : [-\pi, \pi] \to \mathbb{R}$ can be written as a Fourier series

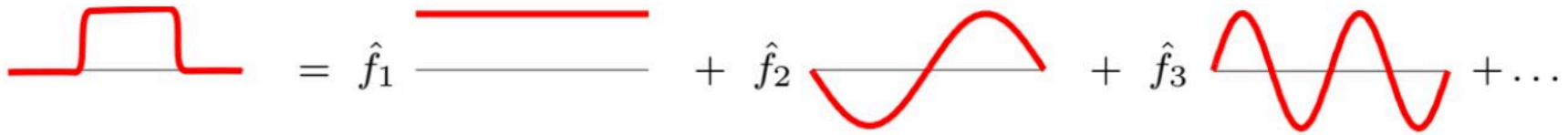
$$f(x) = \sum_{k \geq 0} \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x') e^{-ikx'} \, dx' \, e^{ikx}$$



Fourier Analysis on Euclidean Spaces

A function $f : [-\pi, \pi] \to \mathbb{R}$ can be written as a Fourier series

$$f(x) = \sum_{k \geq 0} \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x') e^{-ikx'} \, dx' \, e^{ikx}$$

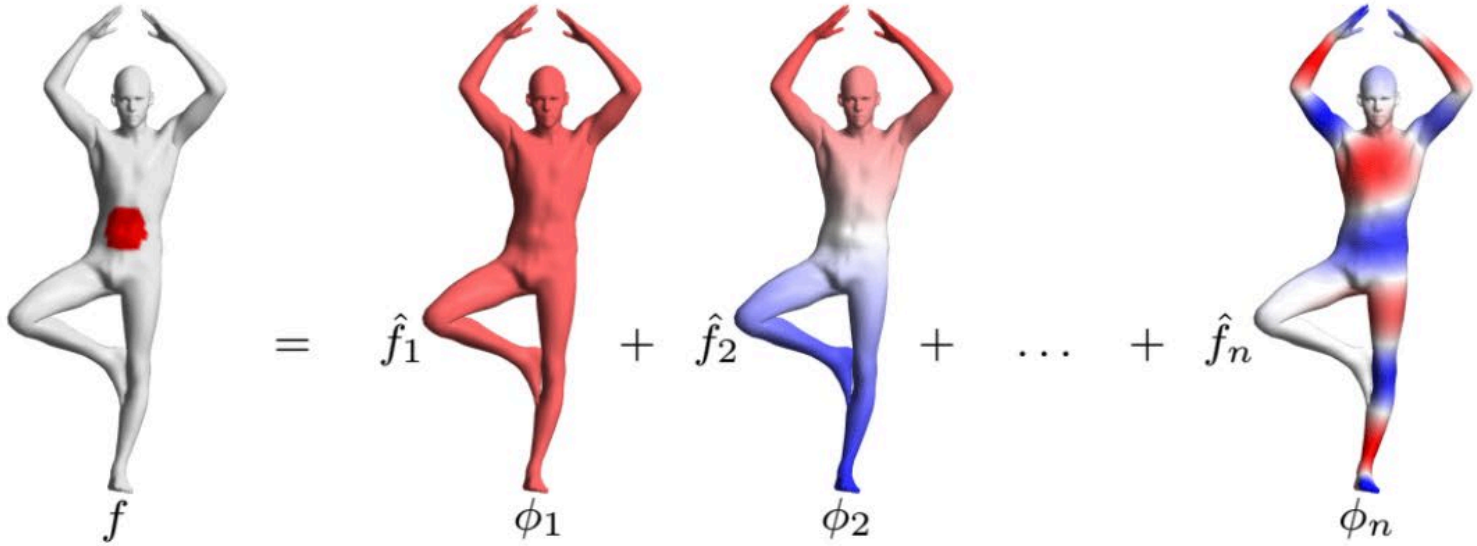


Fourier basis = Laplacian eigenfunctions: $-\frac{d^2}{dx^2} e^{ikx} = k^2 e^{ikx}$

Fourier Analysis on Graphs and Manifolds

A function $f : \mathcal{X} \to \mathbb{R}$ can be written as Fourier series

$$f = \sum_{k=1}^{n} \underbrace{\langle f, \phi_k \rangle_{L^2(\mathcal{X})}}_{\hat{f}_k} \phi_k$$



Fourier basis $=$ Laplacian eigenfunctions: $\Delta\phi_k = \lambda_k \phi_k$

Convolution: Euclidean Space

Given two functions $f, g : [-\pi, \pi] \to \mathbb{R}$ their convolution is a function

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x - x')dx'$$

- Shift-invariance: $f(x - x_0) \star g(x) = (f \star g)(x - x_0)$
- Convolution theorem: Fourier transform diagonalizes the convolution operator $\Rightarrow$ convolution can be computed in the Fourier domain as

$$\widehat{(f \star g)} = \hat{f} \cdot \hat{g}$$

Convolution: Euclidean Space

Convolution of two vectors $\mathbf{f} = (f_1, \ldots, f_n)^\top$ and $\mathbf{g} = (g_1, \ldots, g_n)^\top$

$$\mathbf{f} \star \mathbf{g} = \underbrace{\begin{bmatrix} g_1 & g_2 & \cdots & \cdots & g_n \\ g_n & g_1 & g_2 & \cdots & g_{n-1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ g_3 & g_4 & \cdots & g_1 & g_2 \\ g_2 & g_3 & \cdots & \cdots & g_1 \end{bmatrix}}_{\text{circulant matrix}} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}$$

Convolution: Euclidean Space

Convolution of two vectors $\mathbf{f} = (f_1, \ldots, f_n)^\top$ and $\mathbf{g} = (g_1, \ldots, g_n)^\top$

$$\mathbf{f} \star \mathbf{g} = \begin{bmatrix} g_1 & g_2 & \cdots & \cdots & g_n \\ g_n & g_1 & g_2 & \cdots & g_{n-1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ g_3 & g_4 & \cdots & g_1 & g_2 \\ g_2 & g_3 & \cdots & \cdots & g_1 \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}$$

$$= \mathbf{\Phi} \begin{bmatrix} \hat{g}_1 & & \\ & \ddots & \\ & & \hat{g}_n \end{bmatrix} \mathbf{\Phi}^\top \mathbf{f}$$

Convolution: Euclidean Space

Convolution of two vectors $\mathbf{f} = (f_1, \ldots, f_n)^\top$ and $\mathbf{g} = (g_1, \ldots, g_n)^\top$

$$\mathbf{f} \star \mathbf{g} = \begin{bmatrix} g_1 & g_2 & \cdots & \cdots & g_n \\ g_n & g_1 & g_2 & \cdots & g_{n-1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ g_3 & g_4 & \cdots & g_1 & g_2 \\ g_2 & g_3 & \cdots & \cdots & g_1 \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}$$

$$= \mathbf{\Phi} \begin{bmatrix} \hat{g}_1 & & \\ & \ddots & \\ & & \hat{g}_n \end{bmatrix} \begin{bmatrix} \hat{f}_1 \\ \vdots \\ \hat{f}_n \end{bmatrix}$$

Convolution: Euclidean Space

Convolution of two vectors $\mathbf{f} = (f_1, \ldots, f_n)^\top$ and $\mathbf{g} = (g_1, \ldots, g_n)^\top$

$$\mathbf{f} \star \mathbf{g} = \begin{bmatrix} g_1 & g_2 & \cdots & \cdots & g_n \\ g_n & g_1 & g_2 & \cdots & g_{n-1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ g_3 & g_4 & \cdots & g_1 & g_2 \\ g_2 & g_3 & \cdots & \cdots & g_1 \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}$$

$$= \mathbf{\Phi} \begin{bmatrix} \hat{f}_1 \cdot \hat{g}_1 \\ \vdots \\ \hat{f}_n \cdot \hat{g}_n \end{bmatrix}$$

Spectral Convolution

Spectral convolution of $f, g \in L^2(\mathcal{X})$ can be defined by analogy

$$f \star g \quad = \quad \sum_{k \geq 1} \langle f, \phi_k \rangle_{L^2(\mathcal{X})} \langle g, \phi_k \rangle_{L^2(\mathcal{X})} \phi_k$$

Spectral Convolution

Spectral convolution of $f, g \in L^2(\mathcal{X})$ can be defined by analogy

$$f \star g = \underbrace{\sum_{k \geq 1} \underbrace{\langle f, \phi_k \rangle_{L^2(\mathcal{X})} \langle g, \phi_k \rangle_{L^2(\mathcal{X})}}_{\text{product in the Fourier domain}} \phi_k}_{\text{inverse Fourier transform}}$$

Spectral Convolution

Spectral convolution of $f, g \in L^2(\mathcal{X})$ can be defined by analogy

$$f \star g \quad = \quad \sum_{k \geq 1} \langle f, \phi_k \rangle_{L^2(\mathcal{X})} \langle g, \phi_k \rangle_{L^2(\mathcal{X})} \, \phi_k$$

In matrix-vector notation

$$\mathbf{f} \star \mathbf{g} = \mathbf{\Phi} \left(\mathbf{\Phi}^\top \mathbf{g} \right) \circ \left(\mathbf{\Phi}^\top \mathbf{f} \right)$$

Spectral Convolution

Spectral convolution of $f, g \in L^2(\mathcal{X})$ can be defined by analogy

$$f \star g \quad = \quad \sum_{k \geq 1} \langle f, \phi_k \rangle_{L^2(\mathcal{X})} \langle g, \phi_k \rangle_{L^2(\mathcal{X})} \, \phi_k$$

In matrix-vector notation

$$\mathbf{f} \star \mathbf{g} = \mathbf{\Phi} \left(\mathbf{\Phi}^\top \mathbf{g} \right) \circ \left(\mathbf{\Phi}^\top \mathbf{f} \right)$$

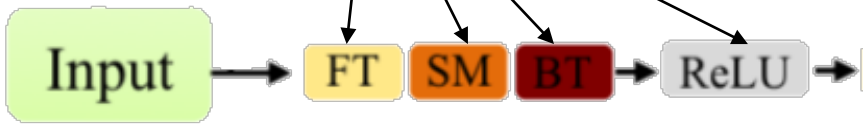
- Not shift-invariant! ($\mathbf{G}$ has no circulant structure)
- Filter coefficients depend on basis $\phi_1, \ldots, \phi_n$

Spectral CNN

Convolutional layer expressed in the spectral domain

$$\mathbf{g}_l = \xi\left(\sum_{l'=1}^{p} \mathbf{\Phi}\hat{\mathbf{W}}_{l,l'}\mathbf{\Phi}^\top \mathbf{f}_{l'}\right) \qquad \begin{matrix} l = 1, \ldots, q \\ l' = 1, \ldots, p \end{matrix}$$

where $\hat{\mathbf{W}}_{l,l} = n \times n$ diagonal matrix of filter coefficients



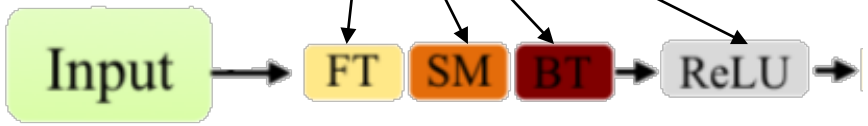
Bruna et al., 2014

Spectral CNN

Convolutional layer expressed in the spectral domain

$$\mathbf{g}_l = \xi\left(\sum_{l'=1}^{p} \mathbf{\Phi}\hat{\mathbf{W}}_{l,l'}\mathbf{\Phi}^\top \mathbf{f}_{l'}\right) \qquad \begin{matrix} l = 1, \ldots, q \\ l' = 1, \ldots, p \end{matrix}$$

where $\hat{\mathbf{W}}_{l,l} = n \times n$ diagonal matrix of filter coefficients



Great! Are we done?

Spectral CNN

Convolutional layer expressed in the spectral domain

$$\mathbf{g}_l = \xi\left(\sum_{l'=1}^{p} \mathbf{\Phi}\hat{\mathbf{W}}_{l,l'}\mathbf{\Phi}^\top \mathbf{f}_{l'}\right) \qquad \begin{matrix} l = 1, \dots, q \\ l' = 1, \dots, p \end{matrix}$$

where $\hat{\mathbf{W}}_{l,l} = n \times n$ diagonal matrix of filter coefficients

- ☹ $\mathcal{O}(n)$ parameters per layer
- ☹ $\mathcal{O}(n^2)$ computation of forward and inverse Fourier transforms $\mathbf{\Phi}^\top, \mathbf{\Phi}$ (no FFT on manifolds or graphs)
- ☹ No guarantee of spatial localization of filters
- ☹ Filters are basis-dependent $\Rightarrow$ does not generalize across domains

Localization and Smoothness

Vanishing moments: In the Euclidean setting

$$\int_{-\infty}^{+\infty} |x|^{2k} |f(x)|^2 dx = \int_{-\infty}^{+\infty} \left| \frac{\partial^k \hat{f}(\omega)}{\partial \omega^k} \right|^2 d\omega$$

Localization in space = smoothness in frequency domain

Localization and Smoothness

Vanishing moments: In the Euclidean setting

$$\int_{-\infty}^{+\infty} |x|^{2k} |f(x)|^2 dx = \int_{-\infty}^{+\infty} \left| \frac{\partial^k \hat{f}(\omega)}{\partial \omega^k} \right|^2 d\omega$$

Localization in space = smoothness in frequency domain

Parametrize the filter using a smooth spectral transfer function $\tau(\lambda)$

Application of the filter

$$\tau(\mathbf{\Delta})\mathbf{f} = \mathbf{\Phi}\tau(\mathbf{\Lambda})\mathbf{\Phi}^\top \mathbf{f}$$

Localization and Smoothness

Vanishing moments: In the Euclidean setting

$$\int_{-\infty}^{+\infty} |x|^{2k} |f(x)|^2 dx = \int_{-\infty}^{+\infty} \left| \frac{\partial^k \hat{f}(\omega)}{\partial \omega^k} \right|^2 d\omega$$

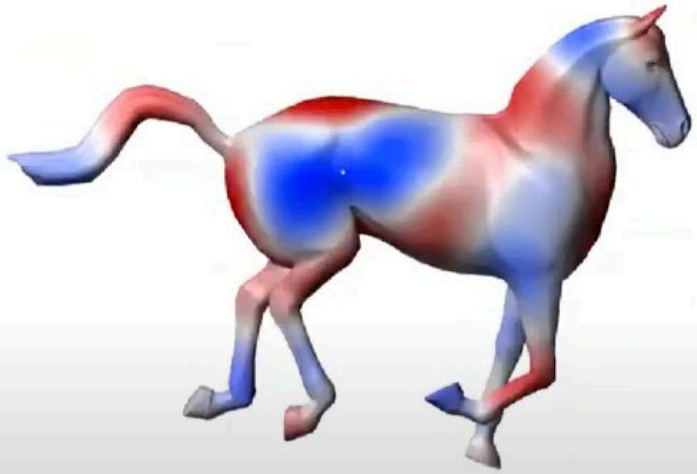
Localization in space = smoothness in frequency domain

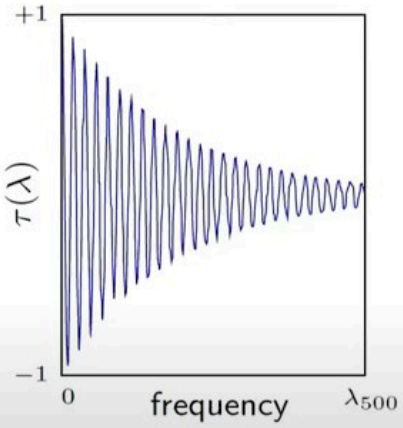
Parametrize the filter using a smooth spectral transfer function $\tau(\lambda)$

Application of the parametric filter with learnable parameters $\boldsymbol{\alpha}$

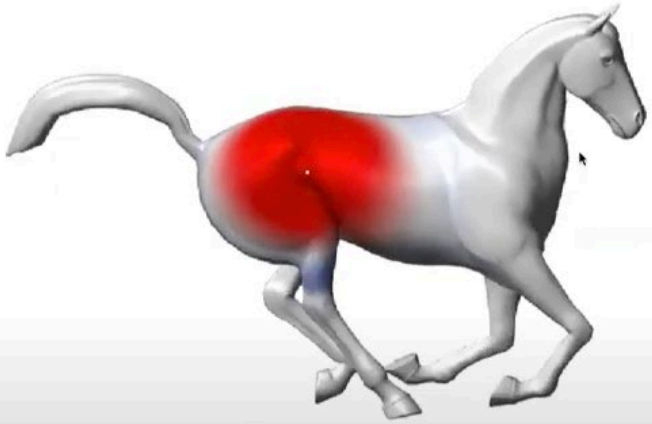
$$\tau_{\boldsymbol{\alpha}}(\boldsymbol{\Delta})\mathbf{f} = \boldsymbol{\Phi} \begin{pmatrix} \tau_{\boldsymbol{\alpha}}(\lambda_1) & & \\ & \ddots & \\ & & \tau_{\boldsymbol{\alpha}}(\lambda_n) \end{pmatrix} \boldsymbol{\Phi}^\top \mathbf{f}$$

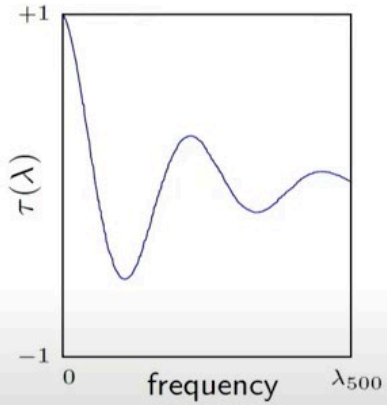
Localization and Smoothness





Non-smooth spectral filter (delocalized in space)





Smooth spectral filter (localized in space)

Spectral gCNN with Polynomial Filters (ChebNet)

Represent spectral transfer function as a polynomial or order r

$$\tau_{\boldsymbol{\alpha}}(\lambda) = \sum_{j=0}^{r} \alpha_j \lambda^j$$

where $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_r)^\top$ is the vector of filter parameters

Spectral gCNN with Polynomial Filters (ChebNet)

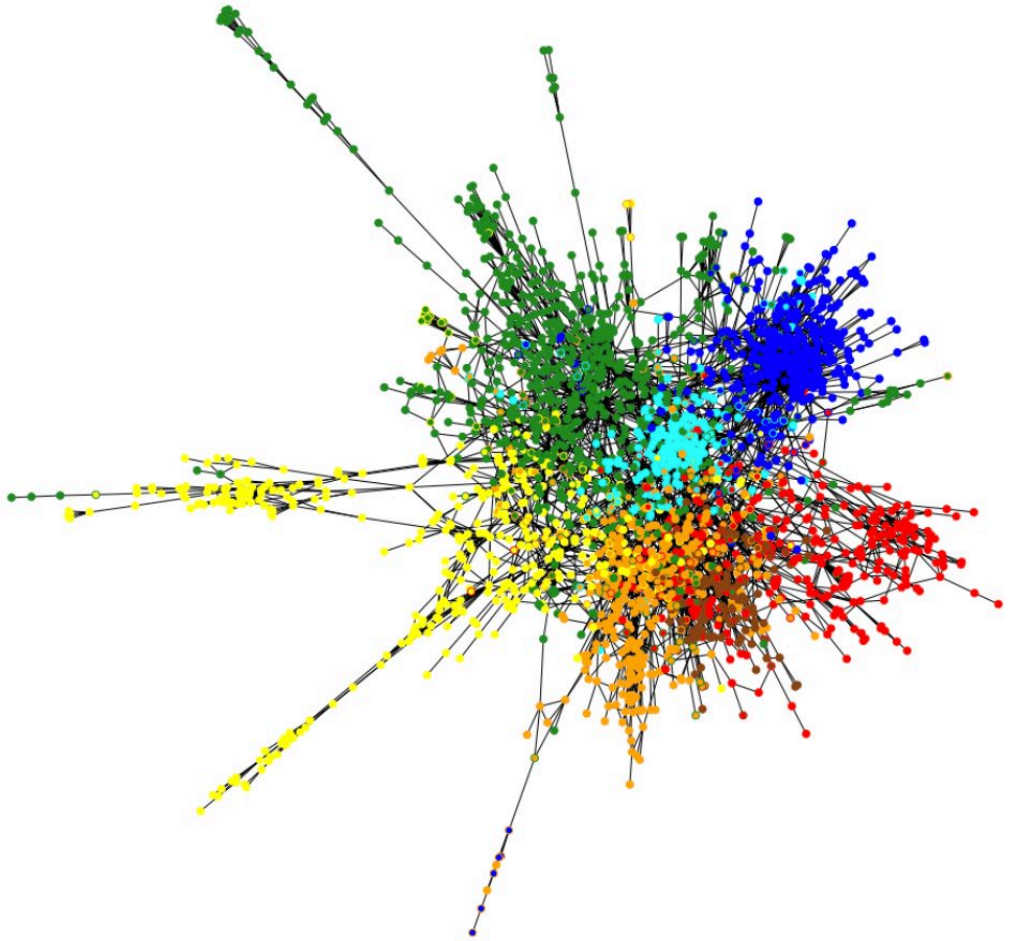
Represent spectral transfer function as a polynomial or order r

$$\tau_{\boldsymbol{\alpha}}(\lambda) = \sum_{j=0}^{r} \alpha_j \lambda^j$$

where $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_r)^\top$ is the vector of filter parameters

- ☺ $\mathcal{O}(1)$ parameters per layer
- ☺ Filters have guaranteed r-hops support
- ☺ No explicit computation of $\boldsymbol{\Phi}^\top, \boldsymbol{\Phi} \Rightarrow \mathcal{O}(nr)$ complexity
- ☹ Does not generalize across domains

Example: Citation Networks



Method	Cora[1]	PubMed[2]
Manifold Regularization[3]	59.5%	70.7%
Semidefinite Embedding[4]	59.0%	71.1%
Label Propagation[5]	68.0%	63.0%
DeepWalk[6]	67.2%	65.3%
Planetoid[7]	75.7%	77.2%
GCN[8]	**81.6%**	**78.7%**

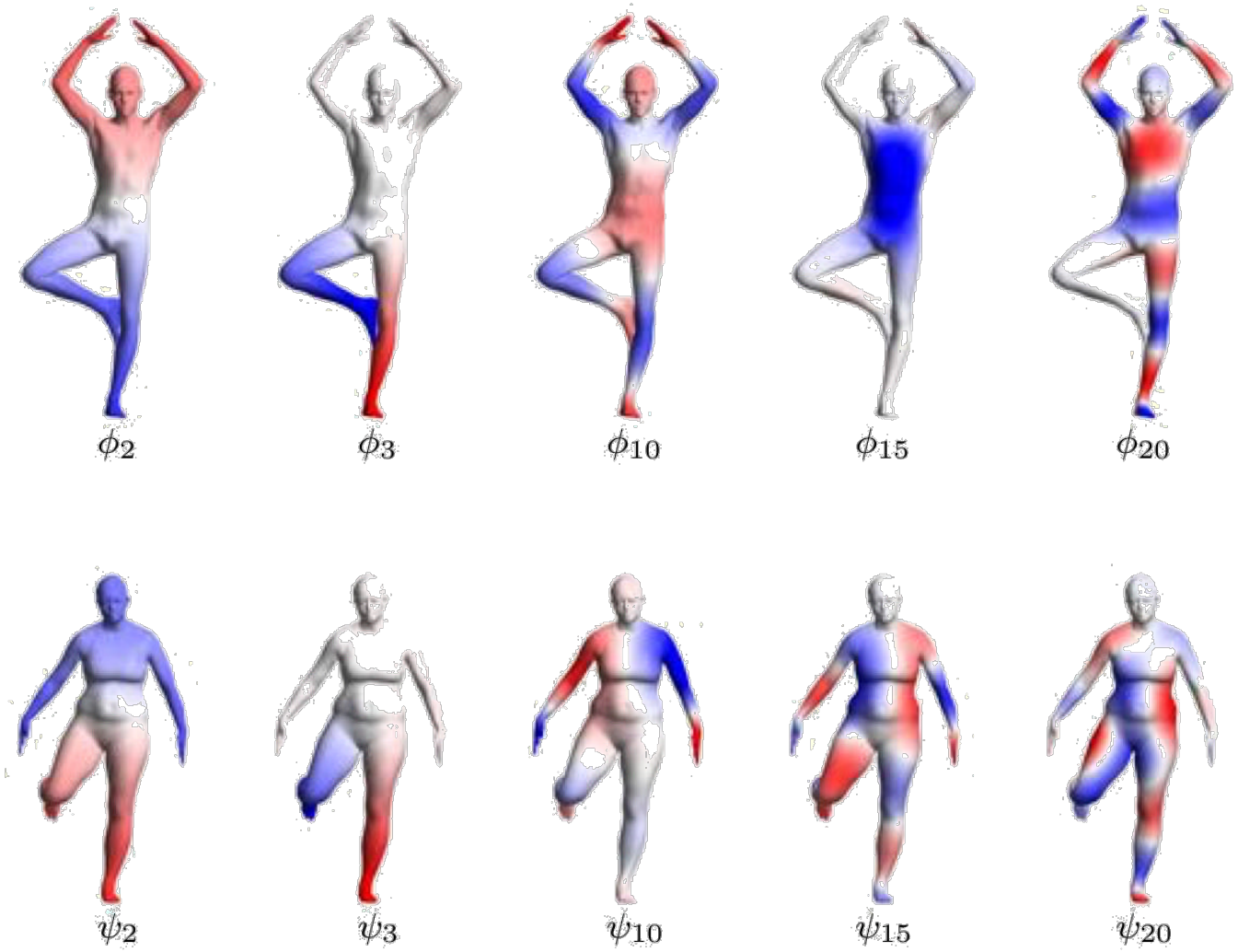
Basis Dependence

Function f

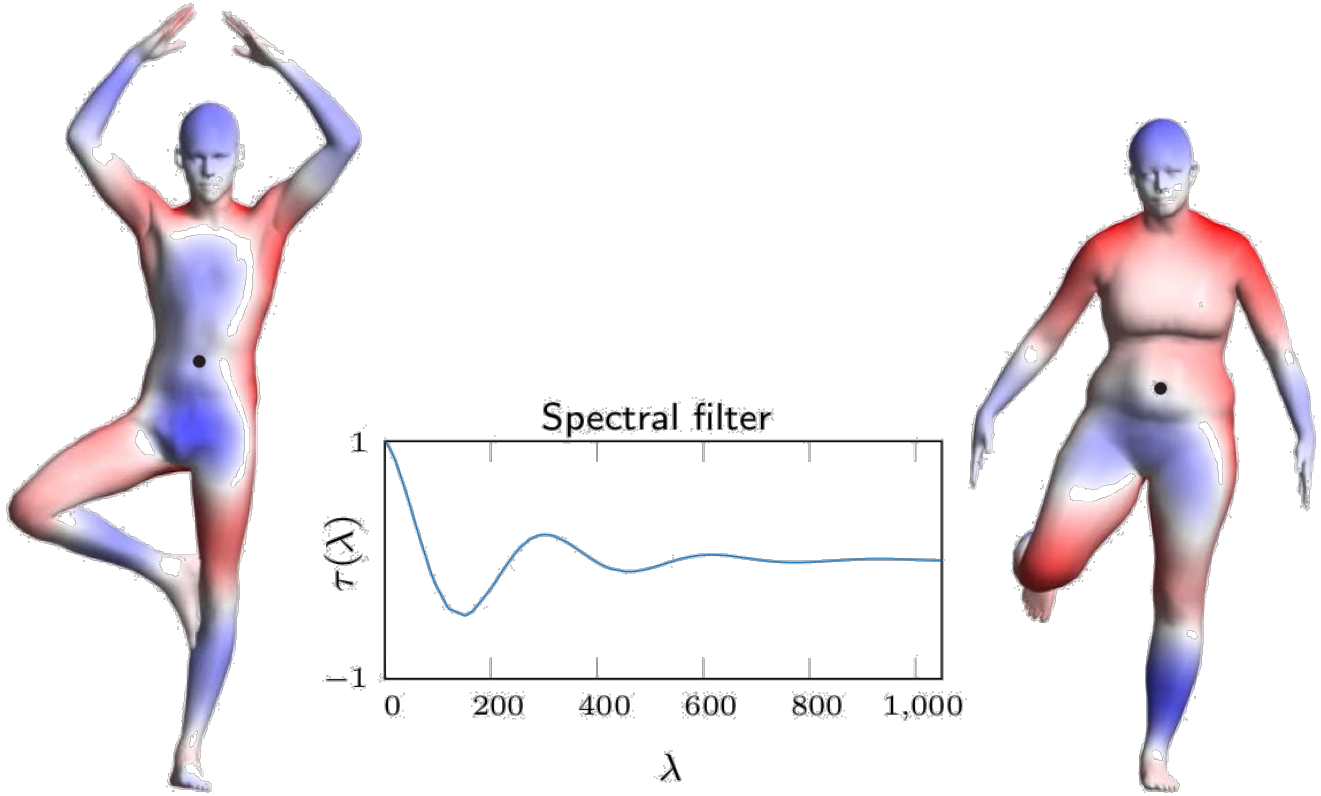
Filtered function $\tilde{f}$

Same function, same filter, another shape

Filtering in Different Bases



Filtering in Different Bases

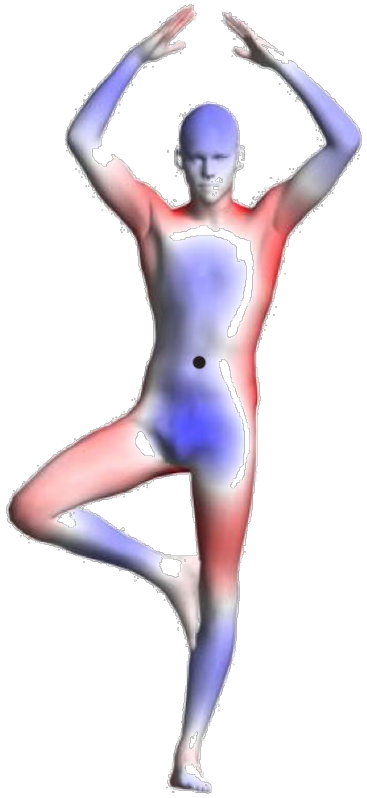


$\boldsymbol{\Phi}\tau(\boldsymbol{\Lambda}_{\boldsymbol{\Phi}})\boldsymbol{\Phi}^\top\boldsymbol{\delta}_0$

$\boldsymbol{\Psi}\tau(\boldsymbol{\Lambda}_{\boldsymbol{\Psi}})\boldsymbol{\Psi}^\top\boldsymbol{\delta}_0$

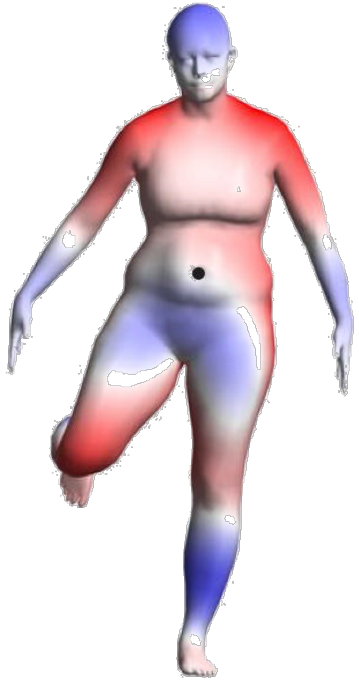
Apply spectral filter $\tau(\lambda)$ in different bases $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$
$\Rightarrow$ **different results!**

Filtering in Different Bases



$\boldsymbol{\Phi}\tau(\boldsymbol{\Lambda}_{\boldsymbol{\Phi}})\boldsymbol{\Phi}^{\top}\boldsymbol{\delta}_0$

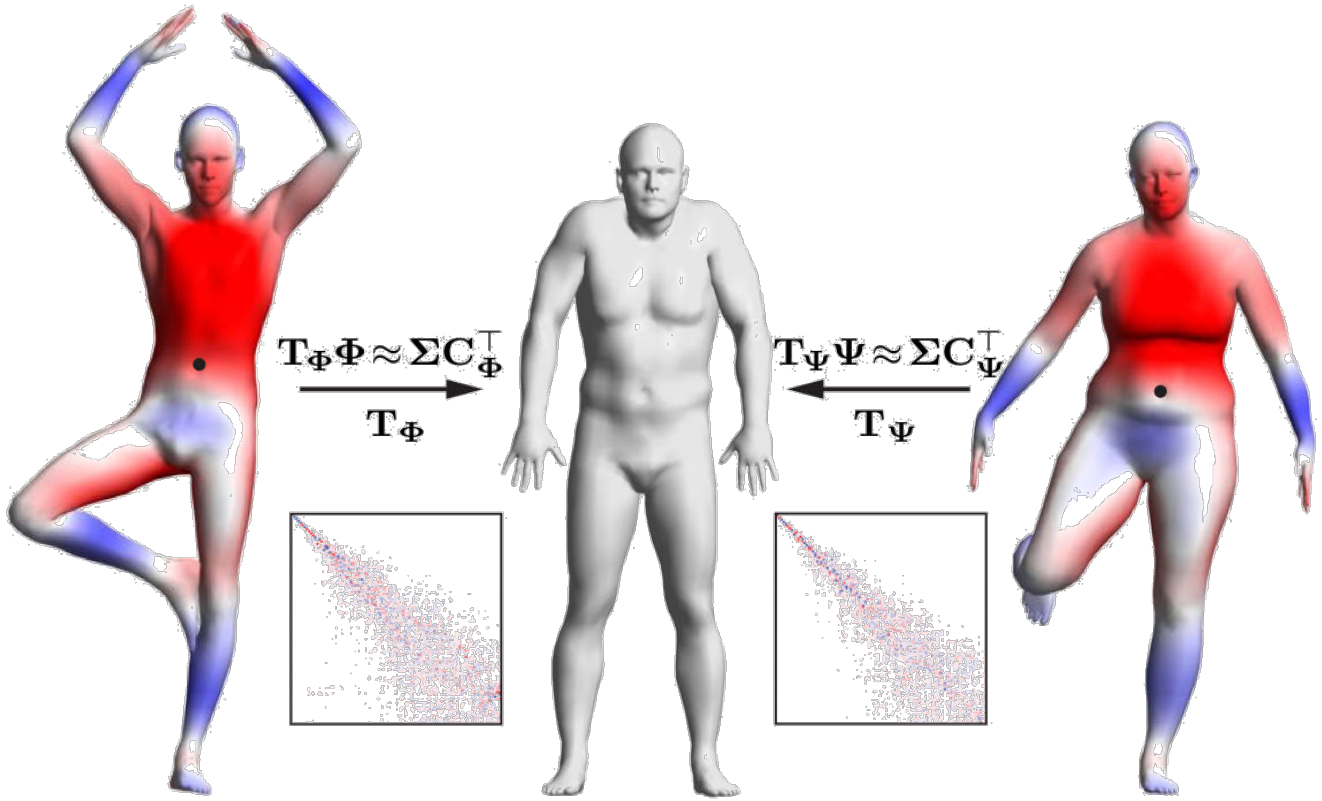
Canonical shape
with basis $\boldsymbol{\Sigma}, \boldsymbol{\Lambda}$



$\boldsymbol{\Psi}\tau(\boldsymbol{\Lambda}_{\boldsymbol{\Psi}})\boldsymbol{\Psi}^{\top}\boldsymbol{\delta}_0$

Apply spectral filter $\tau(\lambda)$ in different bases $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$
$\Rightarrow$ **different results!**

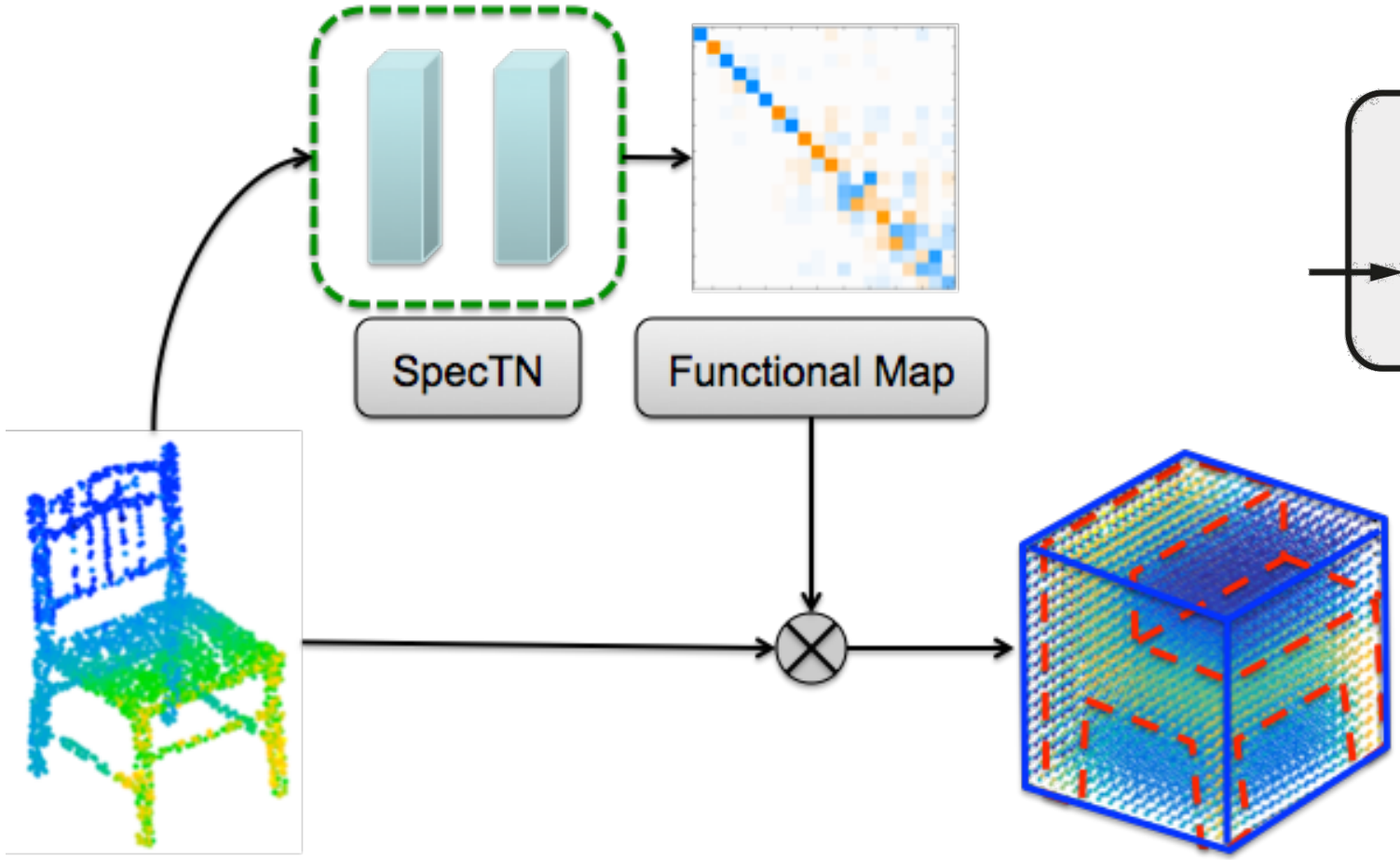
Filtering in Different Bases

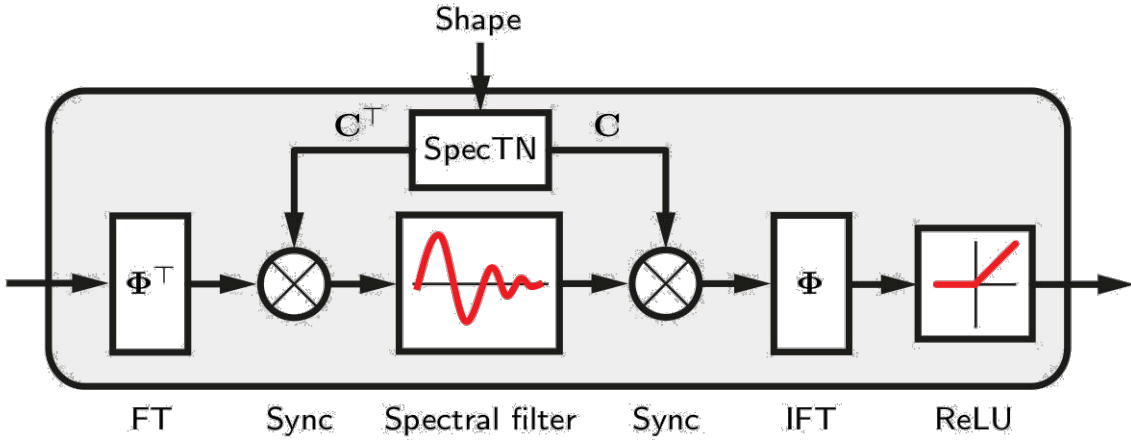


Apply spectral filter $\tau(\lambda)$ in synchronized bases $\Phi \mathbf{C}_{\Phi}$ and $\Psi \mathbf{C}_{\Psi}$
$\Rightarrow$ similar results!

Yi et al. 2017

Spectral Transformer Network



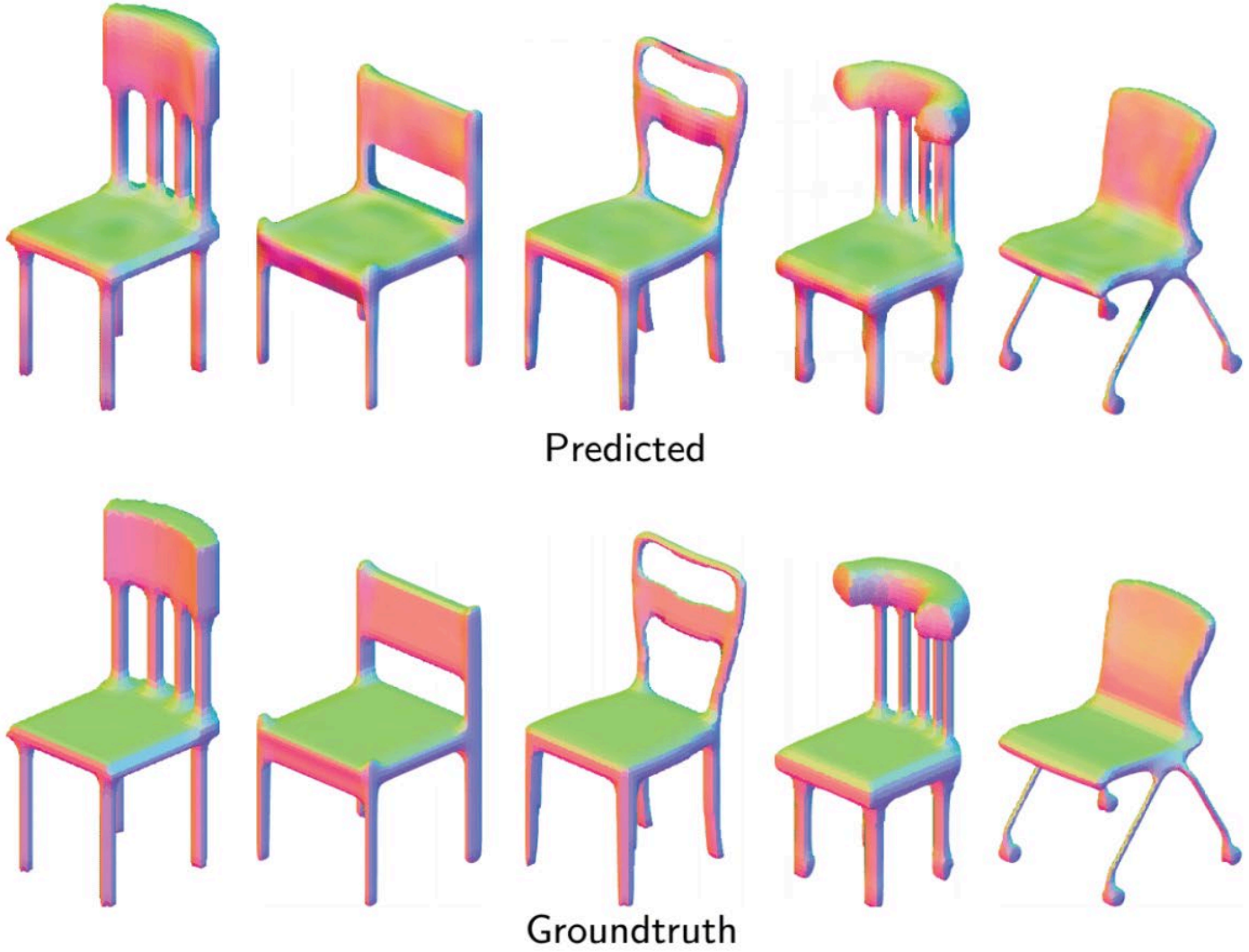


Convolutional filter of a Spectral Transformer Network

Yi et al., 2017

Spectral Transformer Network

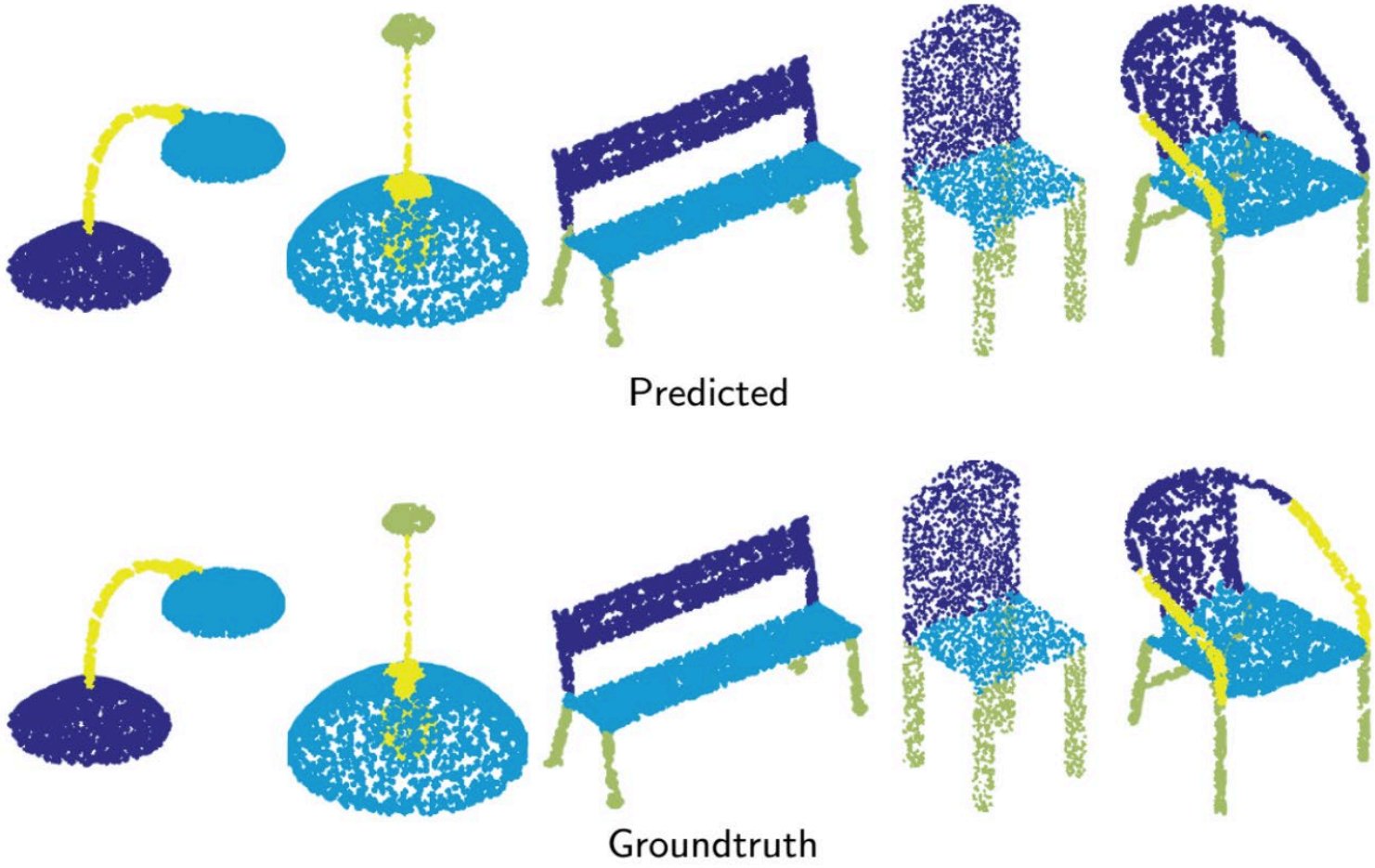
- Normal estimation



Yi et al., 2017

Spectral Transformer Network

- Shape segmentation



Spatial Domain (Charting-based) CNN

Euclidean

Spatial domain

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(x')g(x-x')dx'$$

Spectral domain

$$\widehat{(f \star g)}(\omega) = \hat{f}(\omega) \cdot \hat{g}(\omega)$$

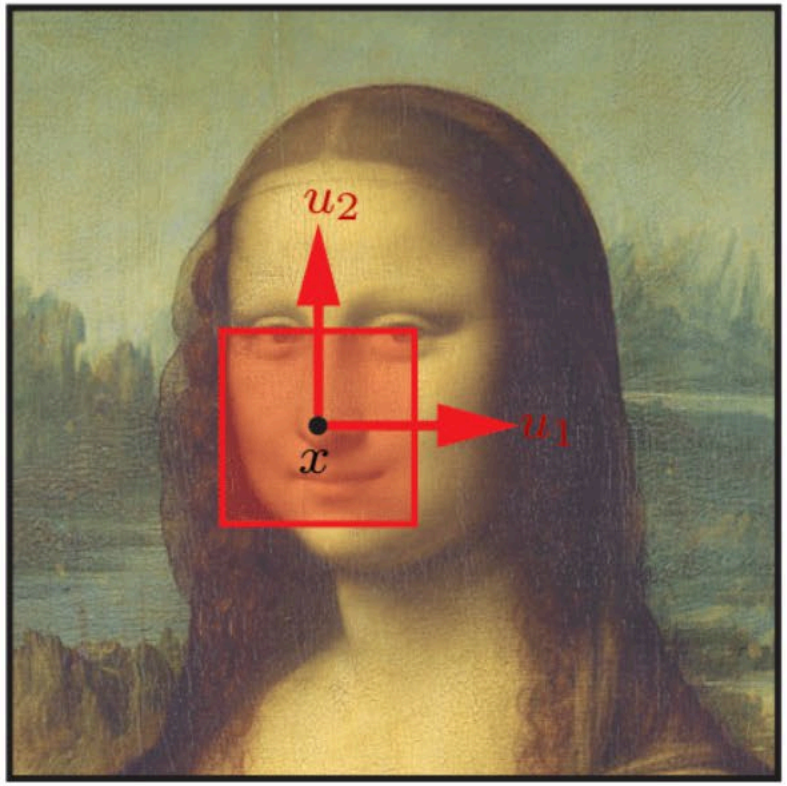
'Convolution Theorem'

Non-Euclidean

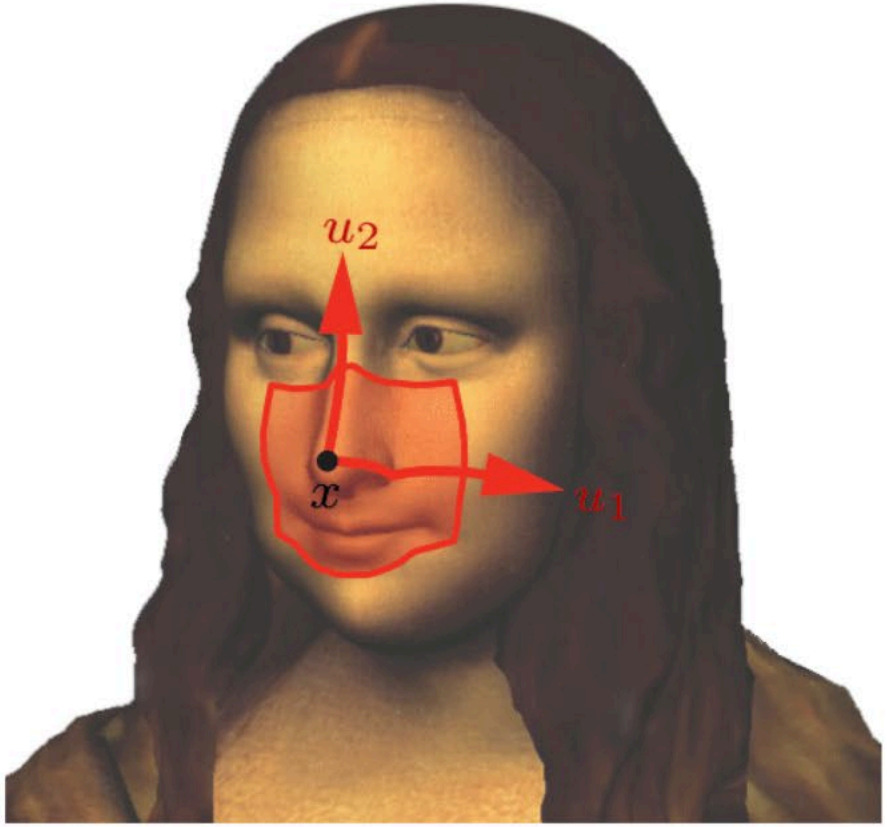
?

$$\widehat{(f \star g)}_k = \langle f, \phi_k \rangle_{L^2(\mathcal{X})} \langle g, \phi_k \rangle_{L^2(\mathcal{X})}$$

Patch Operators

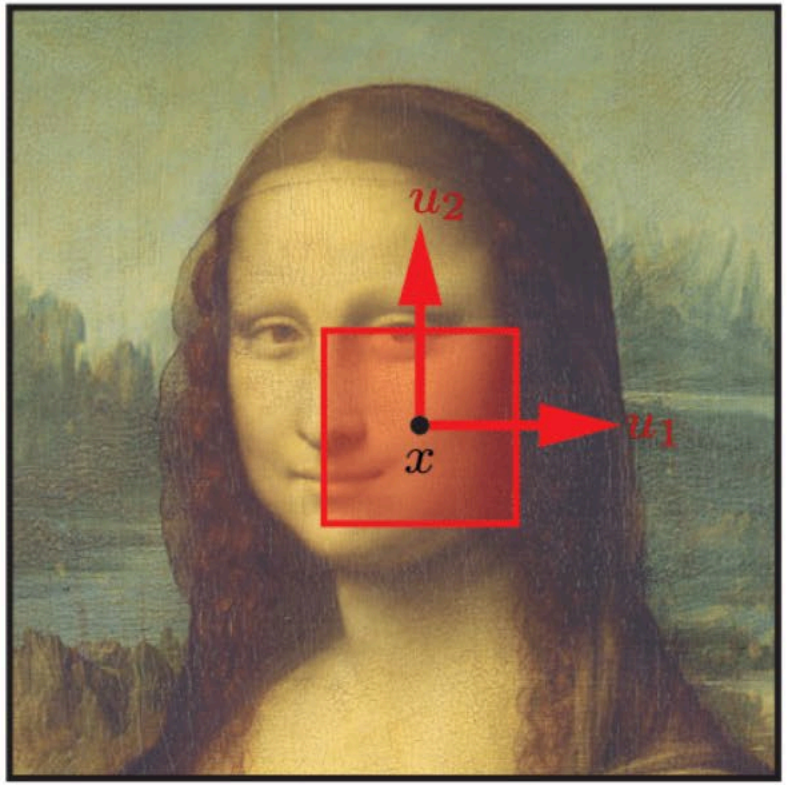


Image

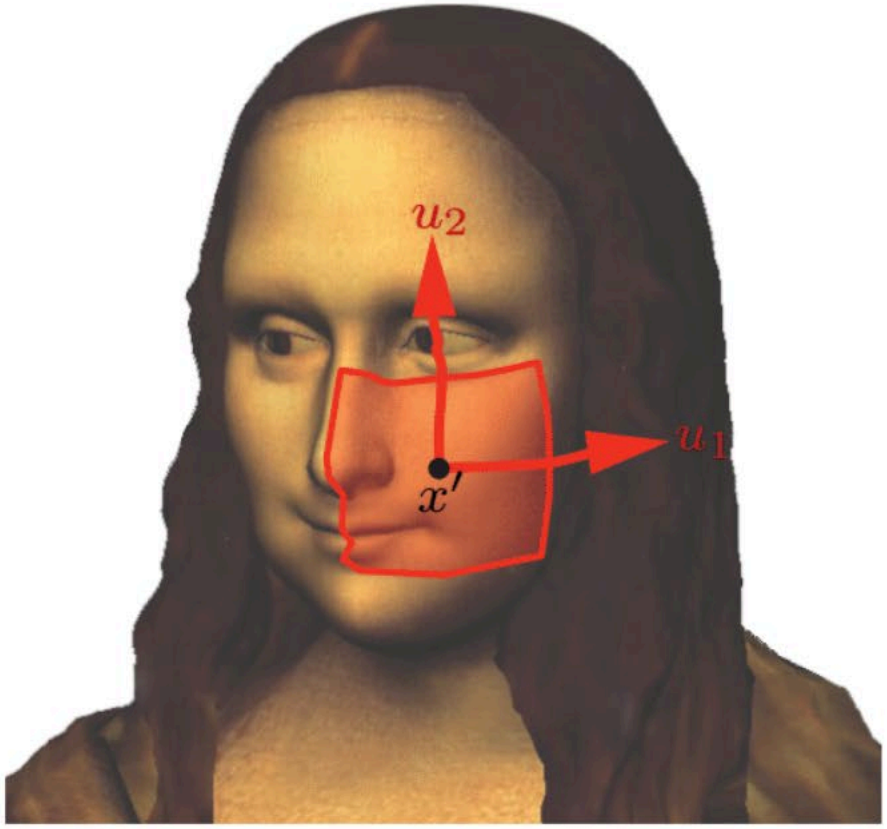


Manifold

Patch Operators



Image



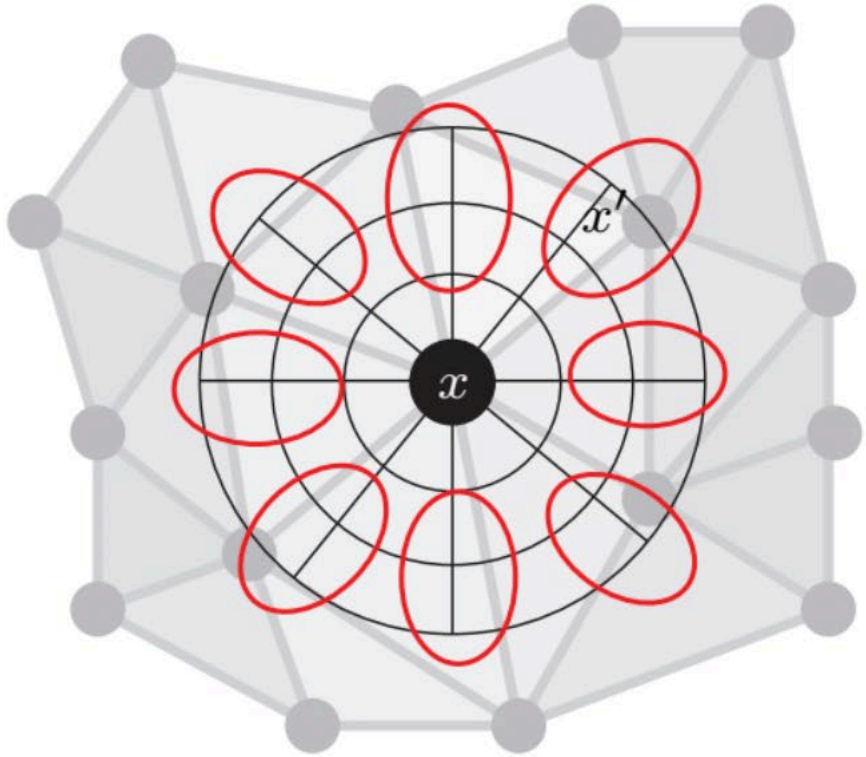
Manifold

Patch Operators

- Local system of coordinates $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)

- Local weights $w_1(\mathbf{u}), \dots, w_L(\mathbf{u})$ w.r.t. $\mathbf{u}$, e.g. Gaussians

$$w_\ell = \exp\left(-(\mathbf{u} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{u} - \boldsymbol{\mu}_\ell)\right)$$



- Spatial convolution with filter g

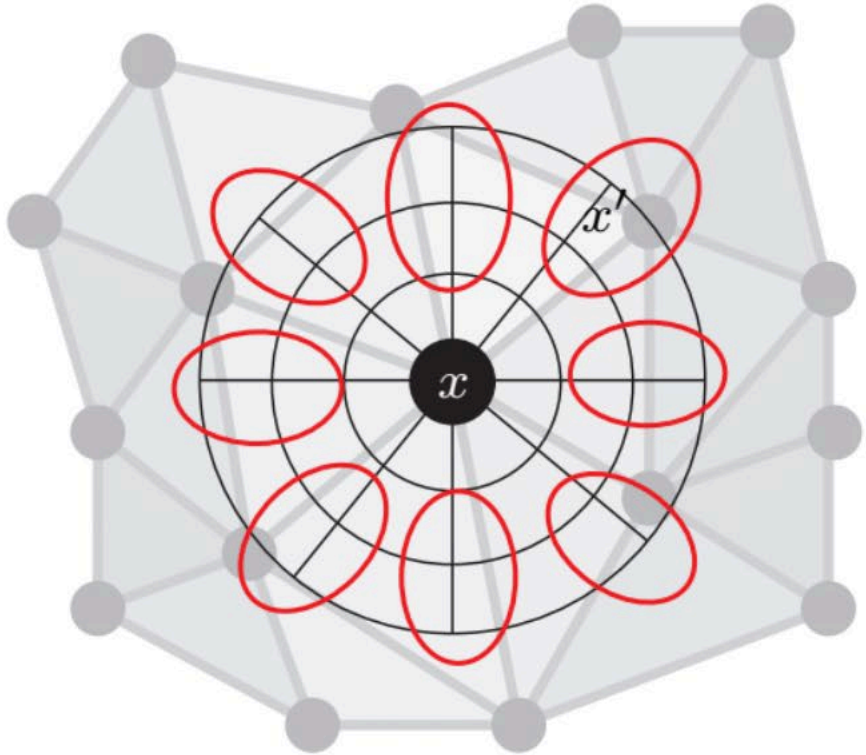
$$(f \star g)(x) \propto \sum_{\ell=1}^{L} g_\ell \int_{\mathcal{X}} w_\ell(\mathbf{u}(x, x')) f(x')\, dx'$$

Patch Operators

- Local system of coordinates $\mathbf{u}(x, x')$ around x' (e.g. geodesic polar)

- Local weights $w_1(\mathbf{u}), \dots, w_L(\mathbf{u})$ w.r.t. $\mathbf{u}$, e.g. Gaussians

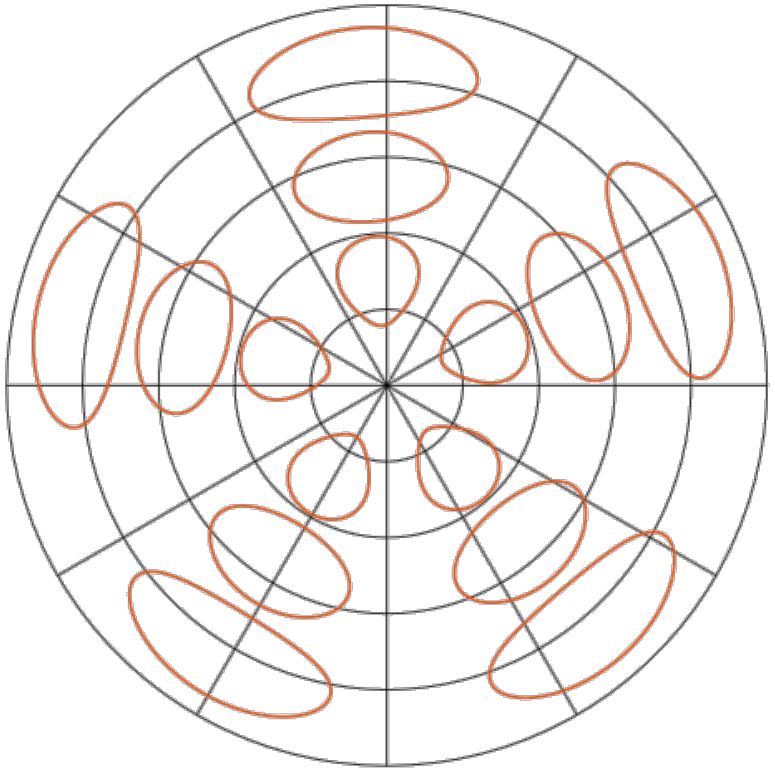
$$w_\ell = \exp\left(-(\mathbf{u} - \boldsymbol{\mu}_\ell)^\top \boldsymbol{\Sigma}_\ell^{-1} (\mathbf{u} - \boldsymbol{\mu}_\ell)\right)$$



- Spatial convolution with filter g

$$(f \star g)(x) \propto \sum_{\ell=1}^{L} g_\ell \underbrace{\int_{\mathcal{X}} w_\ell(\mathbf{u}(x, x')) f(x')\, dx'}_{\text{patch operator}}$$

Geodesic CNN

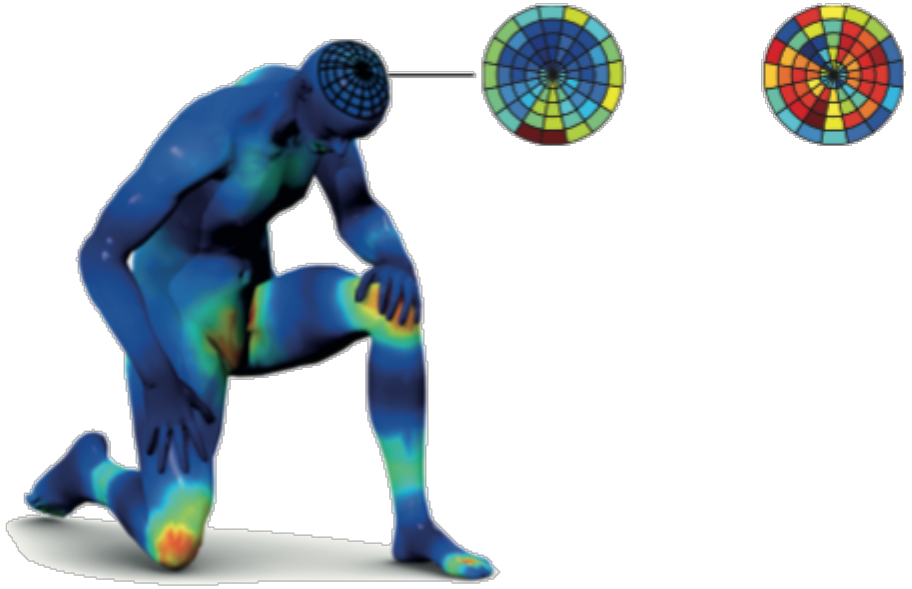


Weighting functions of the geodesic polar patch operator shown in (ρ, θ) coordinates (contours mark the $\frac{1}{2}$-level set)

Geodesic CNN

- Geodesic convolution = apply filter a to patches extracted from $f \in L^2(X)$ in local geodesic polar coordinates

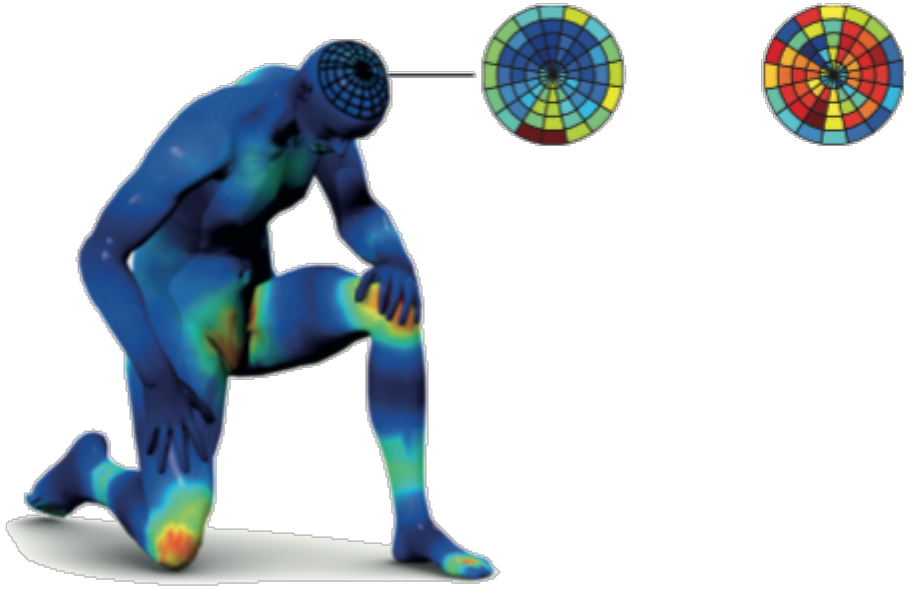
$$(f \star a)(x) = \sum_{\theta,r} \underbrace{(D(x)f)(r,\theta)}\ \underbrace{a(\theta,r)}$$



Geodesic CNN

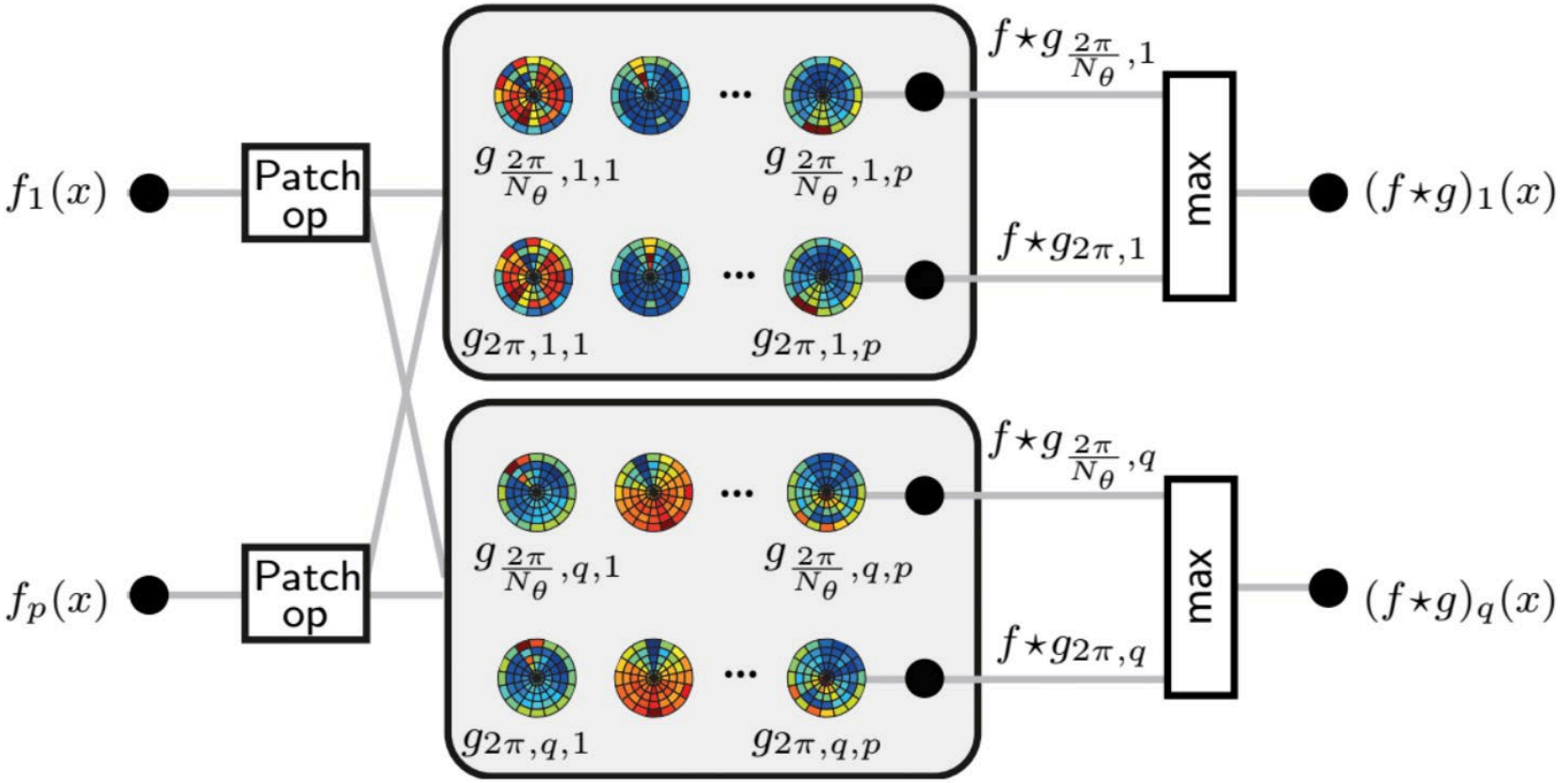
- Geodesic convolution = apply filter a to patches extracted from $f \in L^2(X)$ in local geodesic polar coordinates

$$(f \star a)(x) = \sum_{\theta,r} \underbrace{(D(x)f)(r,\theta)}\ \underbrace{a(\theta,r)}$$



rotation ambiguity

Handling Rotation Ambiguity via Pooling



Conv. layer $$(f_l \star g)_{\Delta\theta,l}(x) = \xi\left(\sum_{\ell=1}^{p}(f_\ell \star g_{\Delta\theta,l,\ell})(x)\right)$$

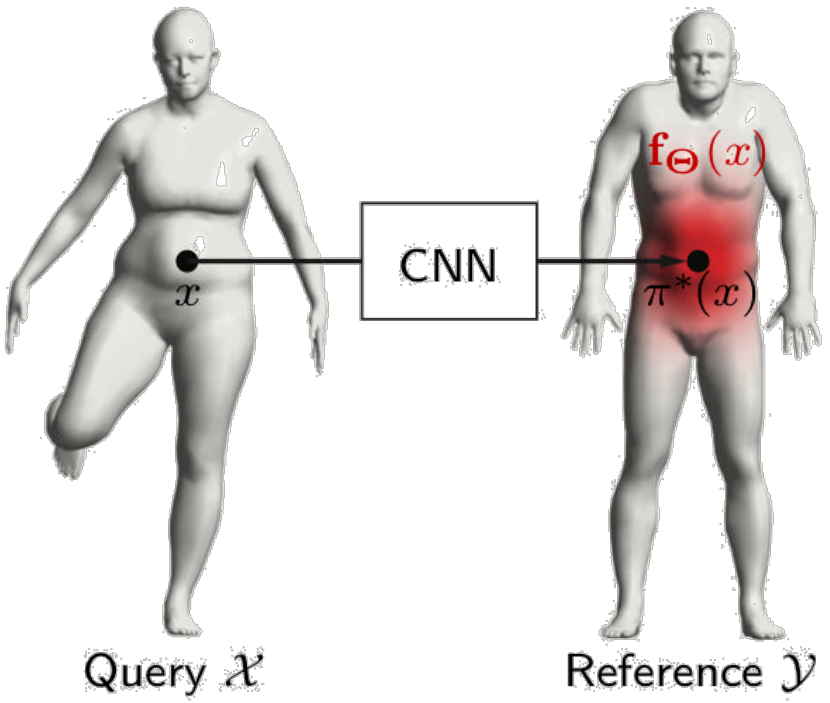
Angular max pooling $$(f \star g)_l(x) = \max_{\Delta\theta}(f \star g)_{\Delta\theta,l}(x)$$

Application: Shape Matching

- Groundtruth correspondence $\pi^* : \mathcal{X} \to \mathcal{Y}$ from query shape $\mathcal{X}$ to some reference shape $\mathcal{Y}$ (discretized with n vertices)
- Correspondence = label each query vertex x as reference vertex y
- Net output at x after softmax layer

$$\mathbf{f}_{\mathbf{\Theta}}(x) = (f_{\mathbf{\Theta},1}(x), \ldots, f_{\mathbf{\Theta},n}(x))$$

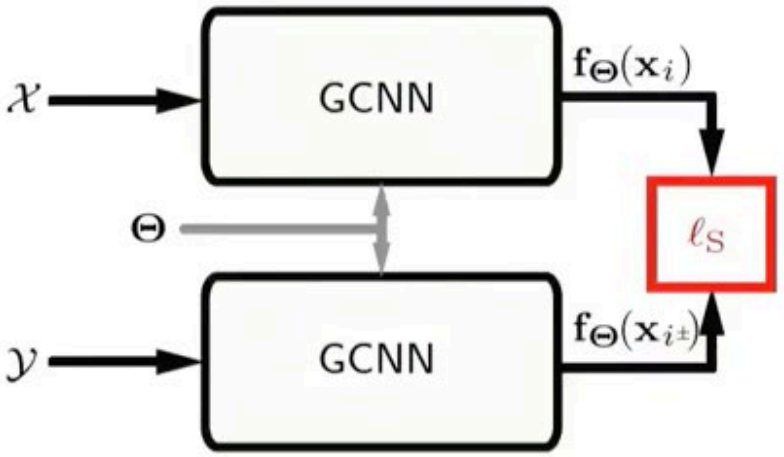
$=$ probability distribution on $\mathcal{Y}$

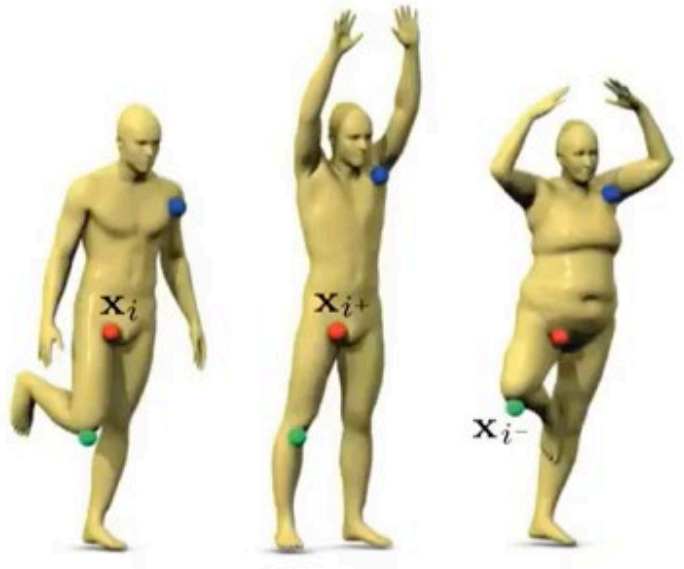


Minimize on training set the cross entropy between groundtruth correspondence and output probability distribution w.r.t. net parameters $\mathbf{\Theta}$

$$\min_{\mathbf{\Theta}} \sum_{x} H(\boldsymbol{\delta}_{\pi*(x)}, \mathbf{f}_{\mathbf{\Theta}}(x))$$

Application: Shape Matching





Training set	positive (i, i^+) and negative (i, i^-) pairs of points
Siamese net	two net instances with shared parameters Θ
Poitwise feature cost	$\ell_S(\Theta) = \gamma \sum_{i,i^+} \|\mathbf{f}_\Theta(\mathbf{x}_i) - \mathbf{f}_\Theta(\mathbf{x}_{i^+})\|_2^2$
	$+ (1-\gamma) \sum_{i,i^-} [\mu - \|\mathbf{f}_\Theta(\mathbf{x}_i) - \mathbf{f}_\Theta(\mathbf{x}_{i^-})\|_2^2]_+$

Application: Shape Matching

Pointwise correspondence error (geodesic distance from groundtruth)

Kim, Lipman, Funkhouser 2011

Application: Shape Matching

Pointwise correspondence error (geodesic distance from groundtruth)

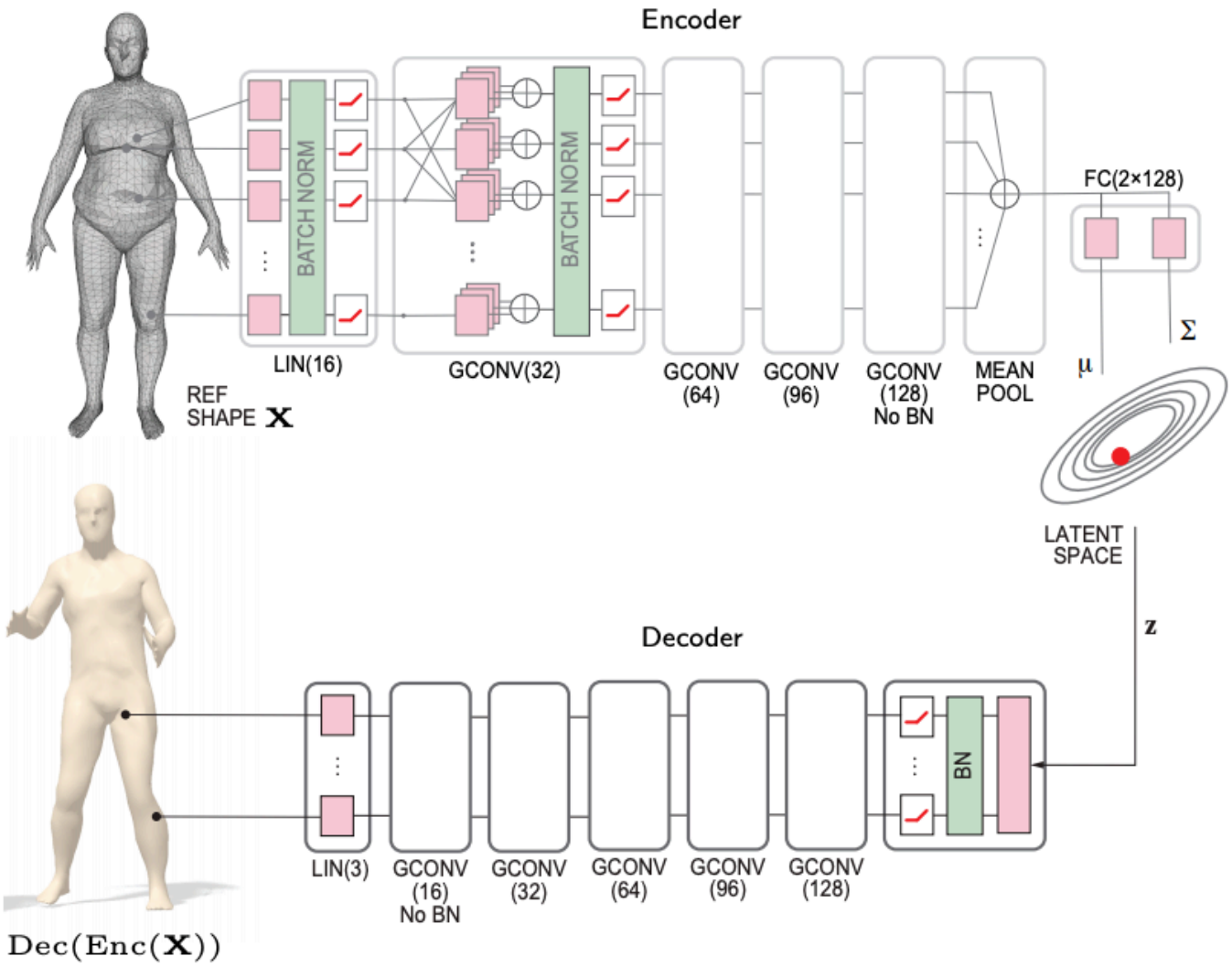
Masci et al. 2015

Application: Shape Completion

Minimize

$$||\mathrm{Dec}(\mathrm{Enc}(\mathbf{X})) - \mathbf{X}||_{\mathrm{F}}$$
$$+\lambda D_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{X})||p(\mathbf{z}))$$

w.r.t. net parameters



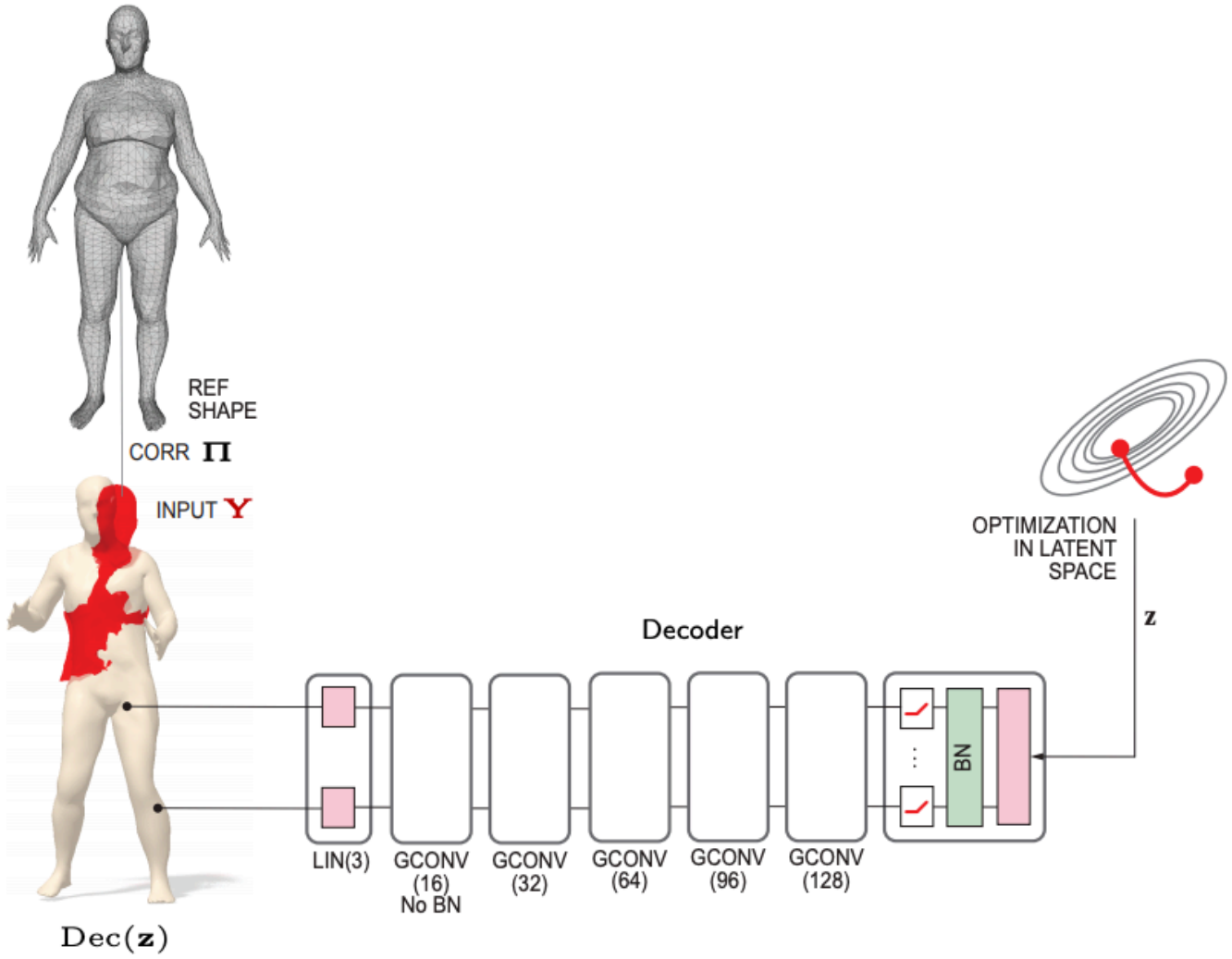
Application: Shape Completion

Minimize

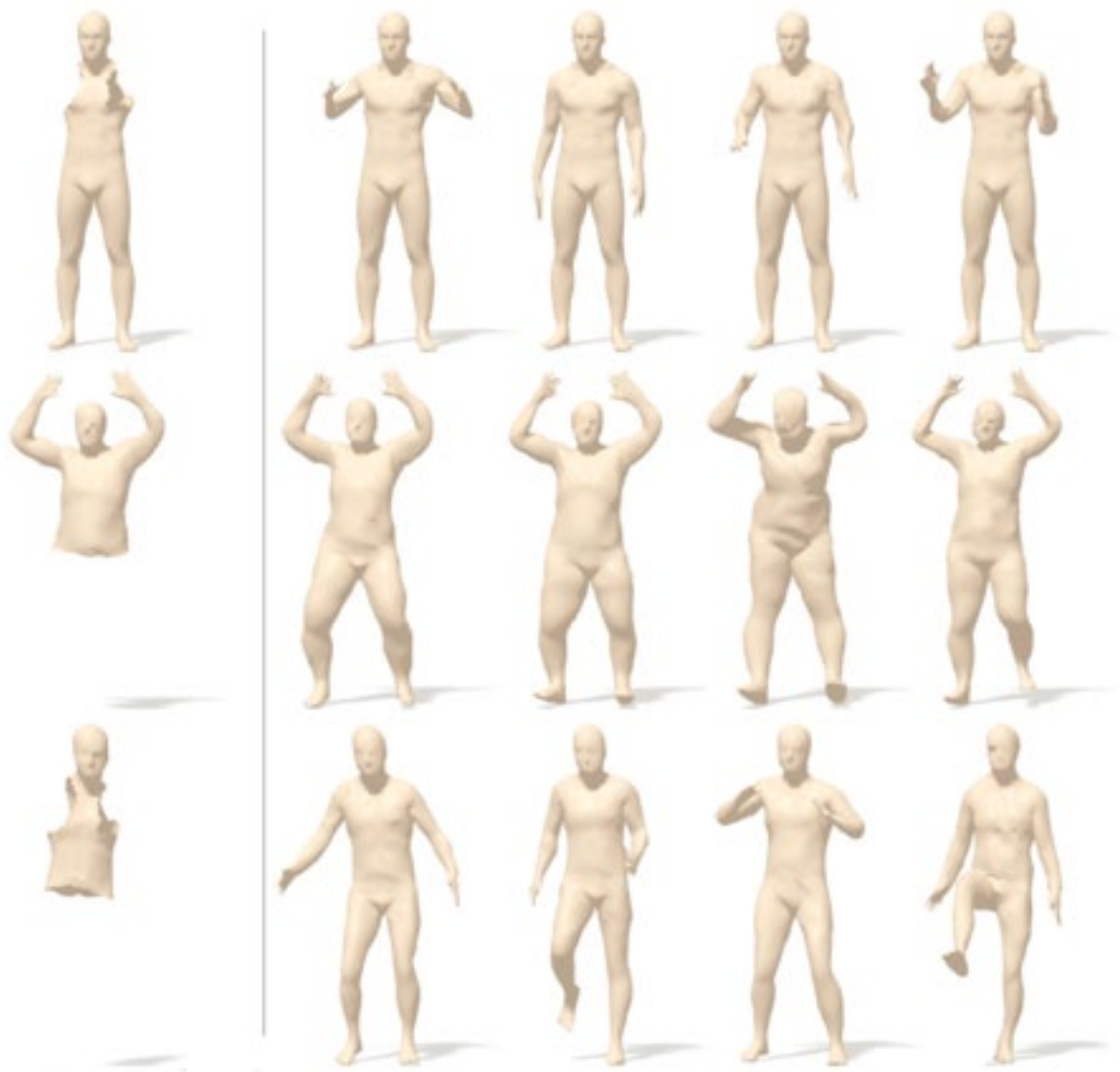
$$\|\mathrm{Dec}(\mathbf{z})\mathbf{\Pi} - \mathbf{T}\mathbf{Y}\|_{\mathrm{F}}$$

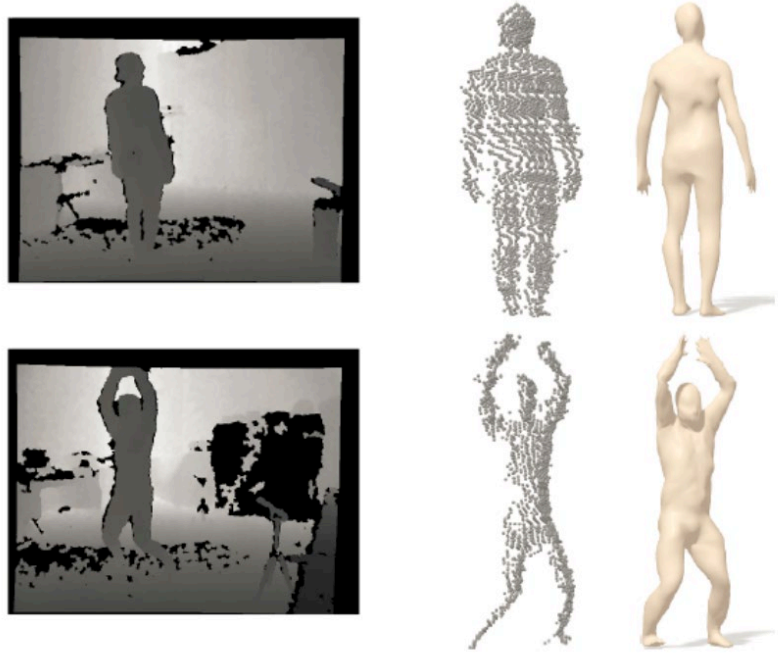
in alternating manner

w.r.t. $\mathbf{z}$ and $\mathbf{T} \in \mathrm{SO}(3)$

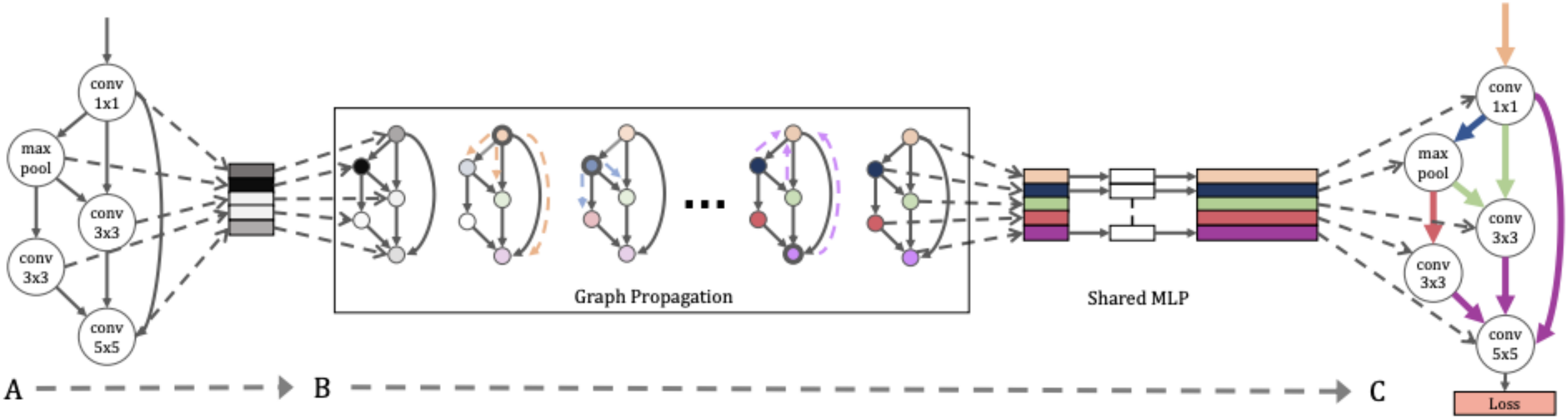


Application: Shape Completion





Graph HyperNets

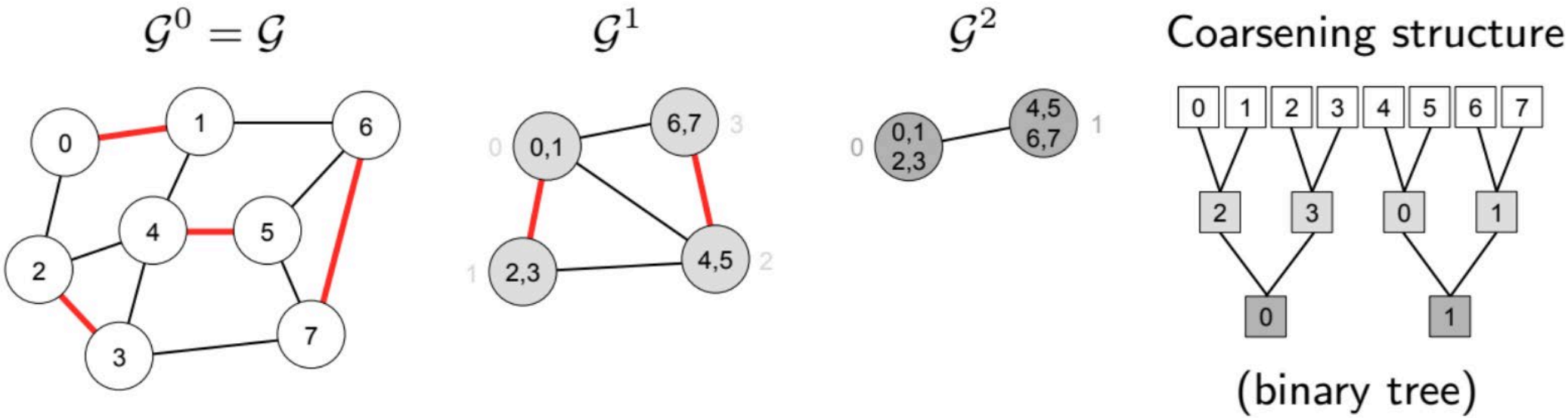


Zhang et al., 2019

Many More Challenges…

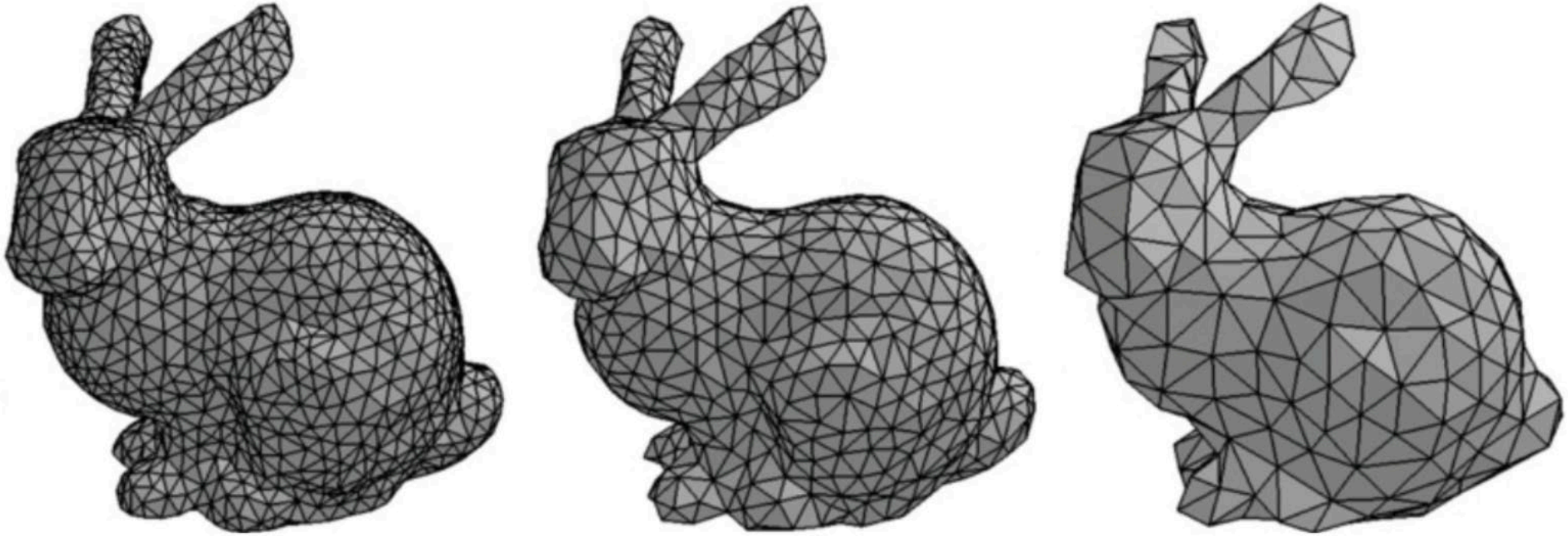
- Pooling
- Input features
- gCNN for edges
- Unknown connectivity
- …

Graph Pooling



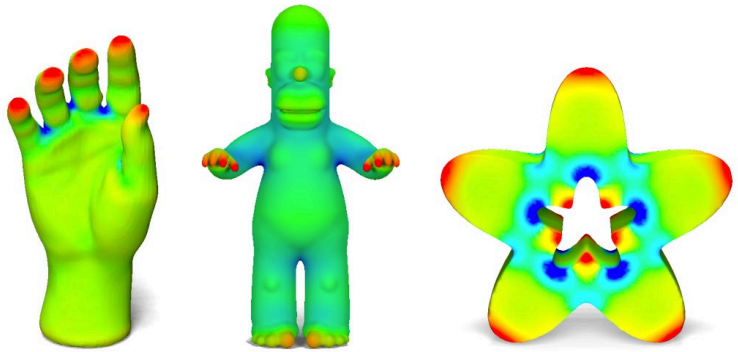
- Produce a sequence of coarsened graphs
- Max or average pooling of collapsed vertices
- Binary tree arrangement of node indices

Mesh Pooling

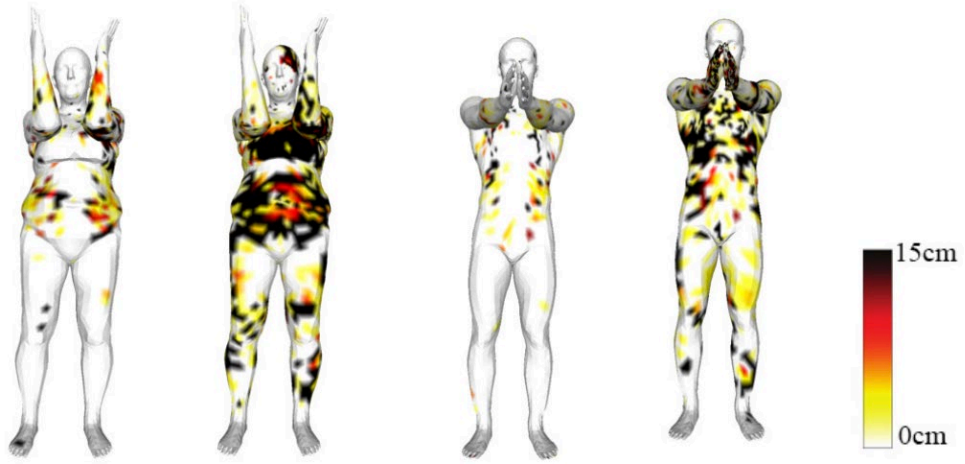


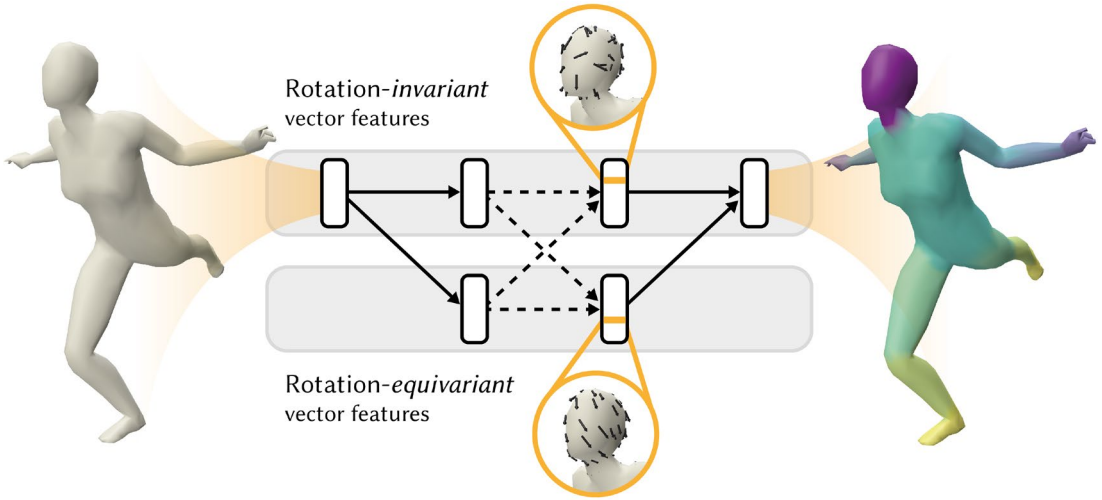
Example of progressive coarsening of a mesh

Features

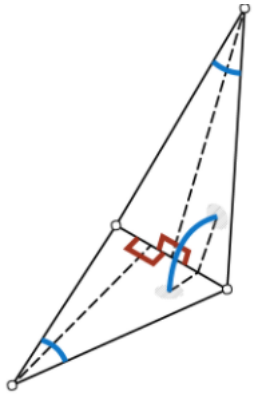


HKS



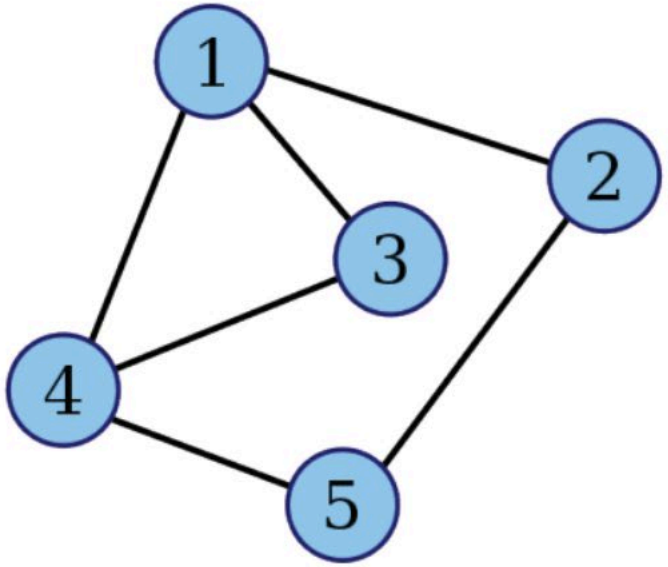


Circular Harmonics



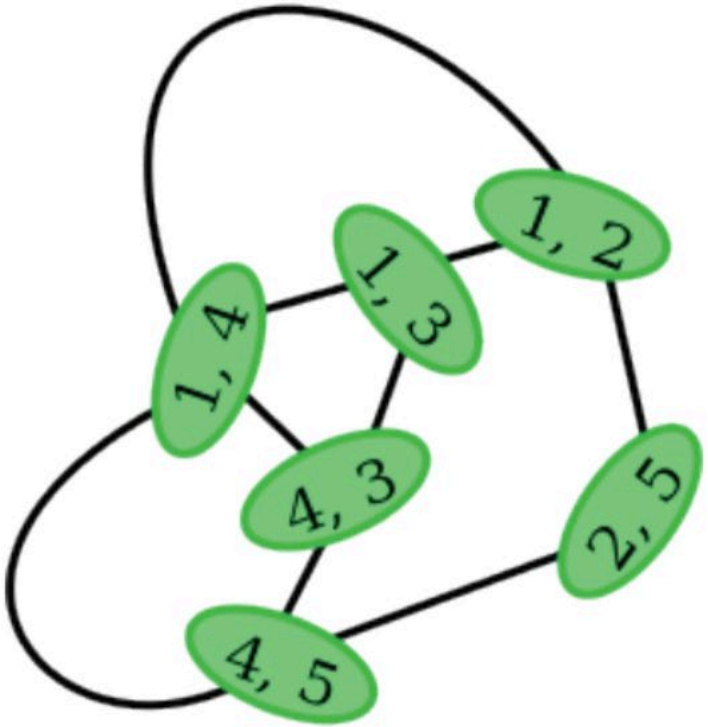
Input Edge Features

Dual-Primal gCNN



Primal graph

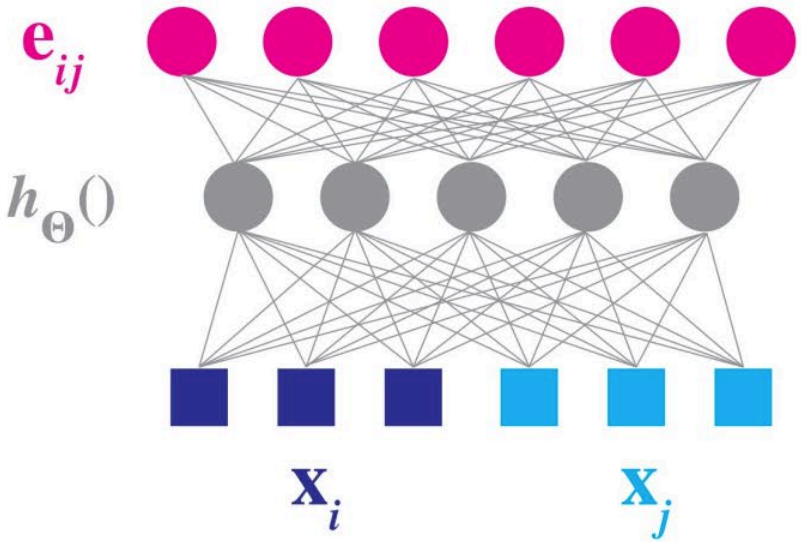
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$

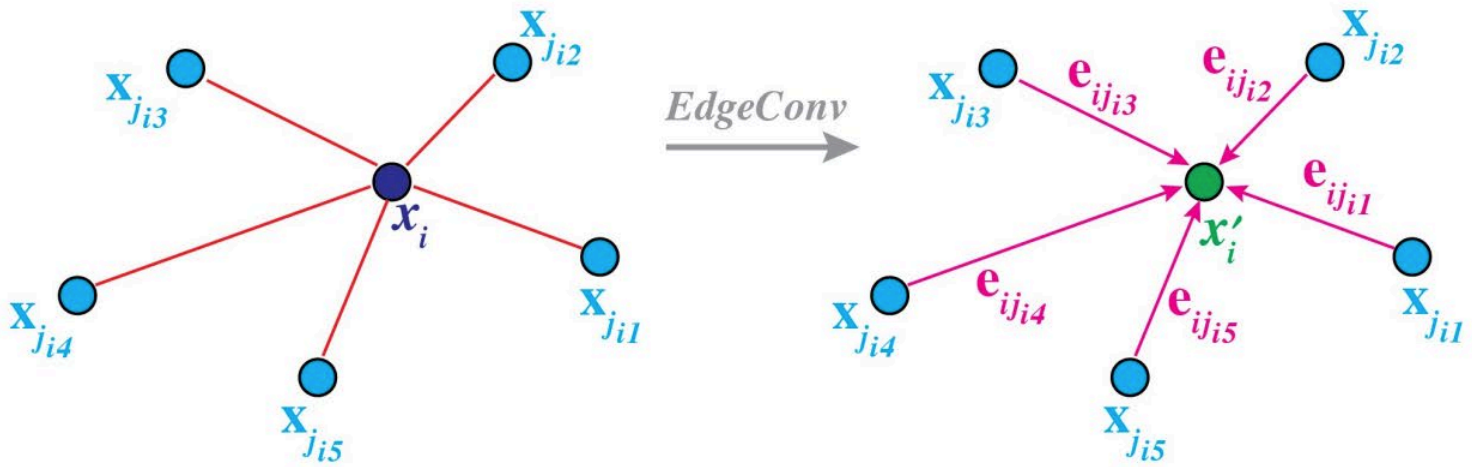


Dual or line graph

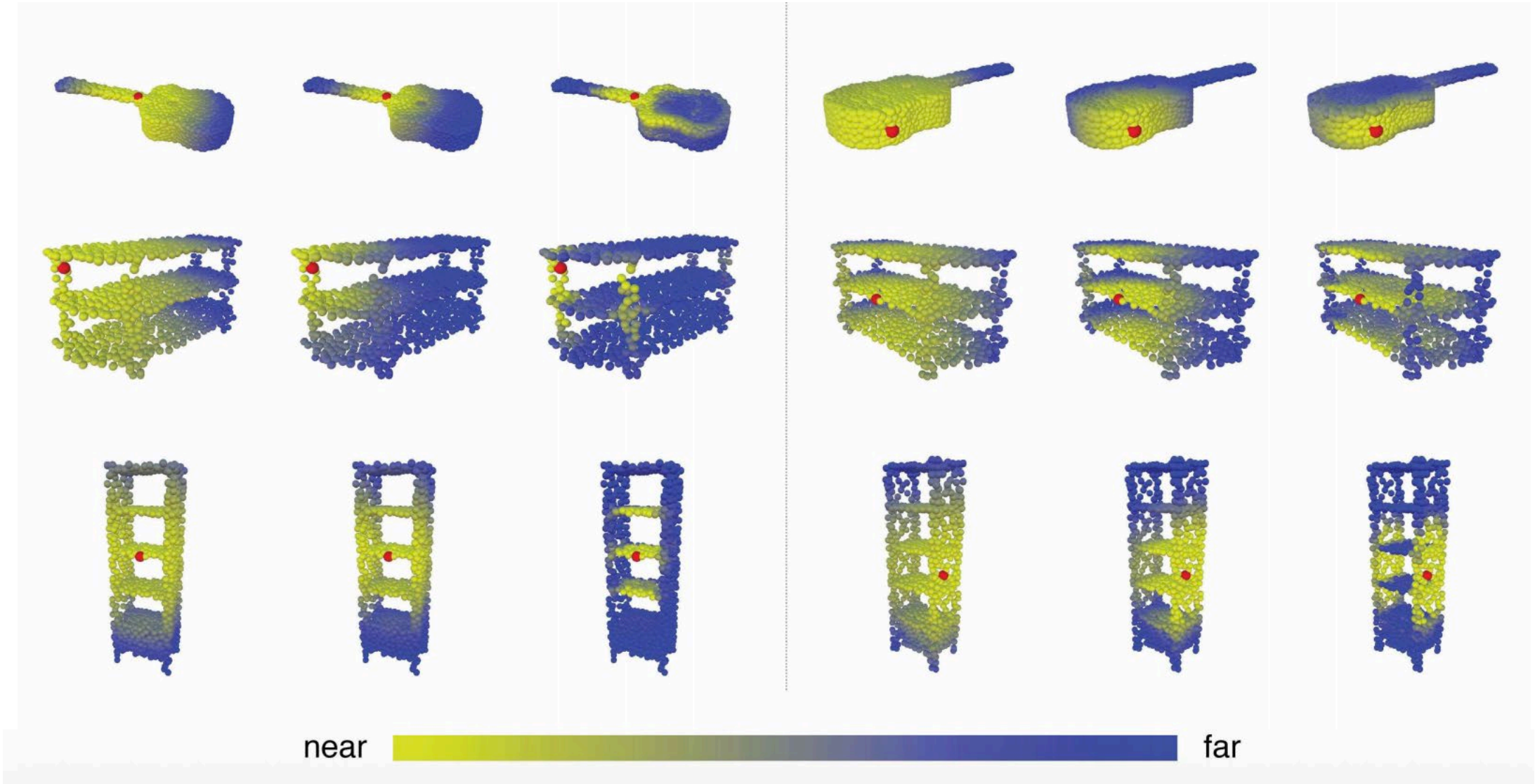
$\tilde{\mathcal{G}} = (\tilde{\mathcal{V}} = \mathcal{E}, \tilde{\mathcal{E}})$

Unknown connectivity (Dynamic Graph CNN)





Unknown Connectivity (Dynamic Graph CNN)



Wang et al., 2018

Discussion

- Spatial construction is usually more efficient but less principled
- Spectral construction is more principled but usually slow (computing Laplacian eigenvectors for large scale data could be painful)
- On going research tries to bridge the gap
- Generalization issue on generic graphs is still a challenge