Learning to Generate Shapes with Geometric Deep Learning

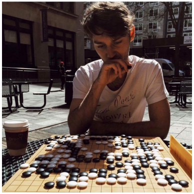
Giorgos Bouritsas

Imperial College London

Minisymposium
on
Distance Metrics and Mass Transfer Between High Dimensional Point Clouds
ICIAM, 17 July 2019, Valencia

Team

G. Bouritsas*
Imperial College London



S. Bokhnyak*
Universita Svizzera Italiana

S. Ploumpis
Imperial College London, FaceSoft.io

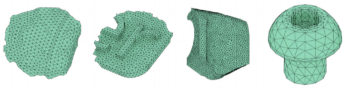
D. Kulon
Imperial College London

S. Zafeiriou
Imperial College London, FaceSoft.io

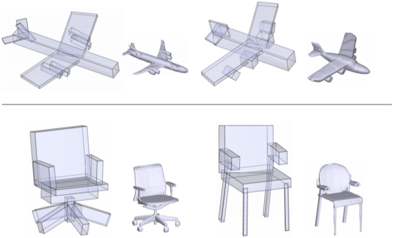
M. Bronstein
Imperial College London, Universita Svizzera Italiana, Twitter

- G. Bouritsas*, S. Bokhnyak* et al., **Neural 3D Morphable Models**, ICCV 2019
- S. Bokhnyak*, G. Bouritsas* et al., **Learning to Represent & Generate Meshes with Spiral Convolutions**, ICLR Workshop, 2019
- D. Kulon, et al., **Single Image 3D Hand Reconstruction with Mesh Convolutions**, BMVC 2019

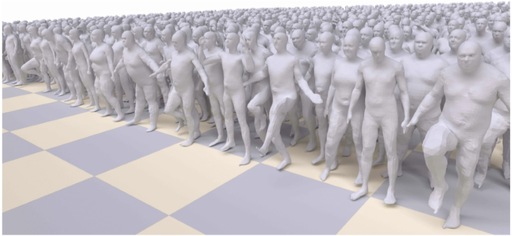
Shape Synthesis



- Engineering & 3D printing



- Computer-aided graphics design

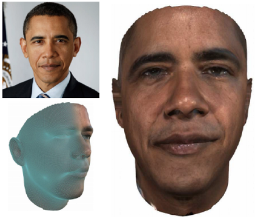


- Synthetic data for ML algorithms training

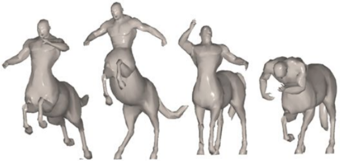
Koch et al., CVPR 2019, Li et al., SIGGRAPH 2017, Ben-Hamu et al., SIGGRAPH ASIA 2018

Representation Learning & Shape priors

- Solving downstream tasks with partially or limited labelled data



- 3D reconstruction



- Shape Classification and Retrieval

Kolotouros et al., CVPR 2019, Gecer et al., CVPR 2019, Bogo et al., CVPR 2017, Bronstein et al., 2008

Challenges: Why is Shape Synthesis so hard?

- **Functionality**: Synthesizing visually pleasing 3D data is not enough: e.g engineering parts need to be highly detailed and functional for real-life use
- **Large dimensionality**: How to make our models scalable?
- **3D acquisition is still not "democratized"**: We still need to deal with limited training data

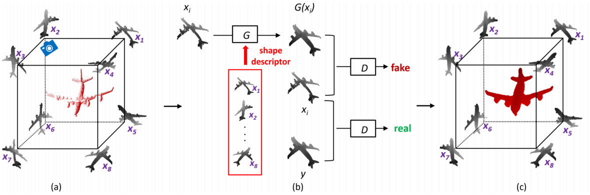
Challenges: Why is Shape Synthesis so hard?

- **Functionality**: Synthesizing visually pleasing 3D data is not enough: e.g engineering parts need to be highly detailed and functional for real-life use
- **Large dimensionality**: How to make our models scalable?
- **3D acquisition is still not "democratized"**: We still need to deal with limited training data
- Ultimate goal: Learn a probability measure over continuous manifolds $\mathbb{P}(\mathcal{X})$

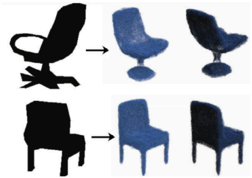
Challenges: Why is Shape Synthesis so hard?

- **Functionality**: Synthesizing visually pleasing 3D data is not enough: e.g engineering parts need to be highly detailed and functional for real-life use
- **Large dimensionality**: How to make our models scalable?
- **3D acquisition is still not "democratized"**: We still need to deal with limited training data
- Ultimate goal: Learn a probability measure over continuous manifolds $\mathbb{P}(\mathcal{X})$
- Relaxing the problem: Learn a probability measure over discretized versions $\Rightarrow$ Multiple representations

Image-based 3D shape generation: Multi-view



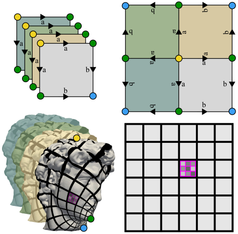
3D shape completion via multi-view depth-maps
Hu et al., arxiv 2019



3D shape synthesis via depth maps and silhouettes
A. Soltani et al., CVPR 2017

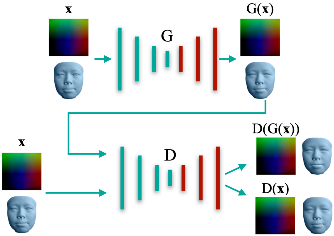
Credit: Anastasia Dubrovina

Image-based 3D shape generation: Mapping to flat domain



- **Seamless Toric Covers**
- **Multi-chart Generation**

Maron et al., SIGGRAPH 2017
Ben-Hamu et al., SIGGRAPH Asia 2018

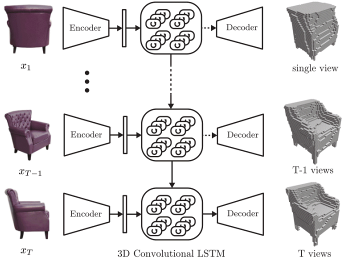


Face Generation using UV maps

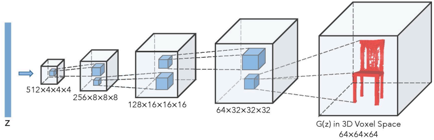
Moschoglou et al., arxiv 2019

Credit: Anastasia Dubrovina

3D Shape Generation via Volumetric Representations



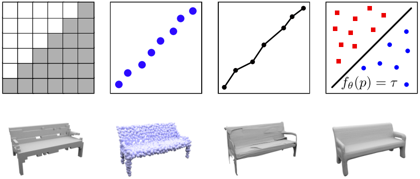
3D Recurrent Reconstruction by multiple views
Choy et al., ECCV 2016



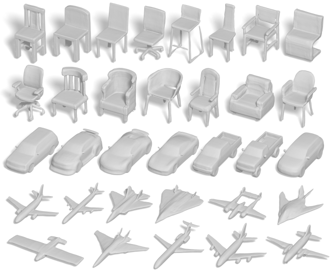
3D-GAN
Wu et al., NIPS 2016

Credit: Anastasia Dubrovina

3D Shape Generation via Implicit Surfaces [CVPR 2019]



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

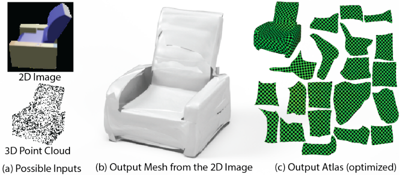


Classify points as exterior or interior of the surface

Michalkiewicz et al., arxiv 2019, Park et al.,
Mescheder et al., Chen and Zhang, CVPR 2019

Credit: Anastasia Dubrovina

3D Shape Generation via Point Clouds



Learnable operators for sets

Achlioptas et al., ICML 2018, Groueix et al., CVPR 2018

Credit: Anastasia Dubrovina

Challenges: Why is Shape Synthesis so hard?

- **Functionality**: Synthesizing visually pleasing 3D data is not enough: e.g engineering parts need to be highly detailed and functional for real-life use
- **Large dimensionality**: How to make our models scalable?
- **3D acquisition is still not "democratized"**: We still need to deal with limited training data
- Ultimate goal: Learn a probability measure over continuous manifolds $\mathbb{P}(\mathcal{X})$
- Relaxing the problem: Learn a probability measure over discretized versions $\Rightarrow$ Multiple representations $\Rightarrow$ Most accurate: **Meshes**

Mesh representation

- ☺ Accurate approximations of the continuous surface
- ☺ Compact and Flexible
- ☺ No post processing needed
- ☹ **Irregularly Structured: Non-euclidean operators needed**

Mesh representation

- ☺ Accurate approximations of the continuous surface
- ☺ Compact and Flexible
- ☺ No post processing needed
- ☹ **Irregularly Structured: Non-euclidean operators needed**

Geometric Deep Learning

aka

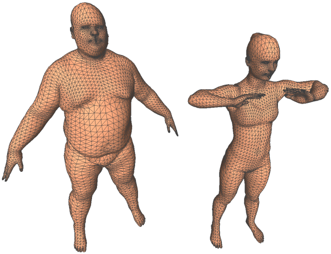
Graph Neural Networks

Mesh representation

- Triangular Mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ with vertices $\mathcal{V} = \{1, \ldots, n\}$, edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, faces $\mathcal{F} \subseteq \mathcal{V} \times \mathcal{V} \times \mathcal{V}$.

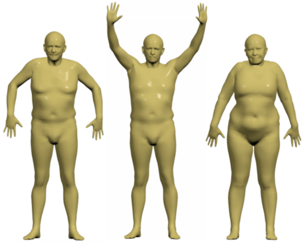
- Signals on the vertices $L^2(\mathcal{V}) = \{\boldsymbol{F} : \mathcal{V} \to \mathbb{R}^d\}$

- Domain: Signals might be defined on a **fixed** or **arbitrary** graph.



Fixed topology

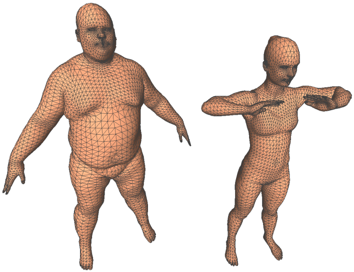
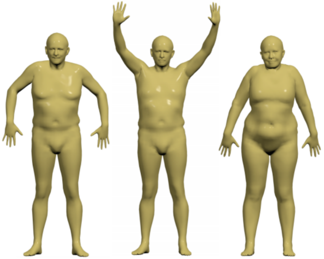
- Unique graph $\mathcal{M}$ for all the shapes s
- Different signal $\boldsymbol{F}_s$ for each shape s



Arbitrary topology

- Different graph $\mathcal{M}_s$ for each shape s
- Different signal $\boldsymbol{F}_s$ for each shape s

Mesh representation

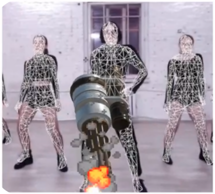
- **Fixed Topology Mesh Generation**: Learn the probability distribution of the signal $\boldsymbol{F}$ that lives on the domain $\mathcal{M}$ (signal generation)
- **Arbitrary Topology Mesh Generation**: Learn the joint probability of the signal $\boldsymbol{F}$ and the domain $\mathcal{M}$ (signal and graph generation)

Fixed Topology Mesh Generation

- Vast amount of 3D data can be represented on the same graph (mesh registration to a template)
- Mainly deformable shapes: Faces, Bodies, Hands etc.
- Applications to 3D reconstruction, animation, VR, AR, etc.

image credit: D. Kulon

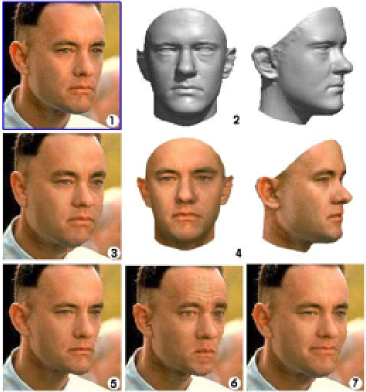
Kanazawa et al., CVPR 2019



www.arielai.com

Fixed Topology Mesh Generation with Statistical Shape Modelling

- Assumption: The signal follows a multi-variate Gaussian distribution.



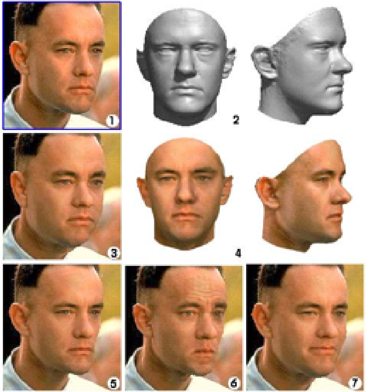
Blanz and Vetter, SIGGRAPH 1999

Fixed Topology Mesh Generation with Statistical Shape Modelling

- Assumption: The signal follows a multi-variate Gaussian distribution.
- Shape model: PCA on the training data
- Let $\boldsymbol{F} \in \mathbb{R}^{n \cdot d}$ the vectorized representation of the signal across the entire mesh. Then:

$$\boldsymbol{F} = \bar{\boldsymbol{F}} + \sum_{i=1}^{k} a_i \sqrt{\lambda_i} \boldsymbol{\phi}_i$$

where $\bar{\boldsymbol{F}}$, the estimated mean, $\boldsymbol{\phi}_i$, λ_i the principal eigenvectors and eigenvalues of the covariance matrix, $a_i \sim \mathcal{N}(0, 1)$.



Blanz and Vetter, SIGGRAPH 1999

Fixed Topology Mesh Generation with Statistical Shape Modelling

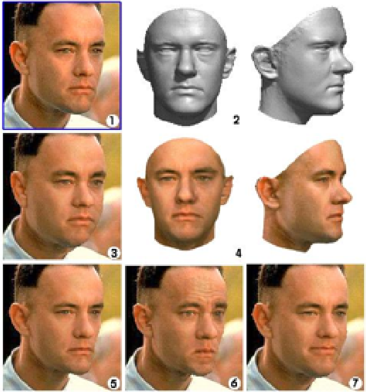
- Assumption: The signal follows a multi-variate Gaussian distribution.

☹ Global: The underlying connectivity of the domain remains unused

☹ Large number of parameters $O(n)$

☹ Linear

☹ Strong assumption (gaussianity)



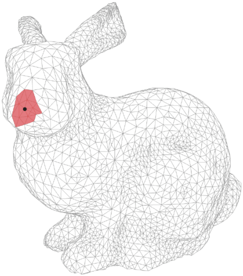
Blanz and Vetter, SIGGRAPH 1999

Define local learnable operators on the underlying graph domain!

Fixed Topology Mesh Generation with Graph Neural Networks

Define local learnable operators on the underlying graph domain!

- ☺ Local: stationarity assumption allows to learn local filters that can be transferred across the domain
- ☺ Reduced number of parameters $O(1)$
- ☺ Non-linear: adding non-linearities between consecutive GNN layers
- ☺ Hierarchical: defining graph pooling operators
- ☺ Minimum assumptions about the distribution needed



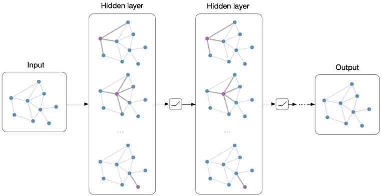


figure by Thomas Kipf

Graph Neural Networks: The Message Passing paradigm

- Every local filter at every layer is equivalent to a message passing operation
- Node features are learned by exchanging information with neighbouring nodes

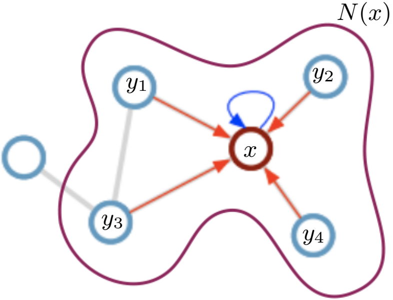


figure by Thomas Kipf

Graph Neural Networks: The Message Passing paradigm

- Every local filter at every layer is equivalent to a message passing operation
- The operation needs to be permutation invariant
- The operation needs to be transferable across different neighborhoods

$$\boldsymbol{F}'(x) = \rho^{\mathcal{E}\to\mathcal{V}}\big(\{\boldsymbol{F}(y)\}_{y\in\mathcal{N}(x)}\big)$$

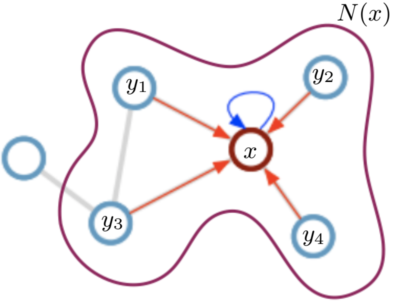


figure by Thomas Kipf

Graph Neural Networks: Spectral Kernels

- First attempts: filters originally defined in the spectral domain (using the convolution theorem).
- $\rho^{\mathcal{E}\to\mathcal{V}}$ parametrised via R-th order graph Laplacian polynomials $T_r(\mathbf{\Delta})$.

$$\mathbf{F}' = \xi\left(\sum_{r=0}^{R} T_r(\mathbf{\Delta})\mathbf{F}\,\boldsymbol{G}_r\right)$$

- GCN (Kipf et al., ICLR 2017): $k=1$, i.e. only immediate neighbours are taken into account $\mathbf{F}' = \xi\big(T_1(\mathbf{\Delta})\mathbf{F}\boldsymbol{G}\big)$. Following the message passing notation:

$$\mathbf{F}'(x) = \xi\left(\sum_{y\in\mathcal{N}(x)} T_1(\mathbf{\Delta})\mathbf{F}(y)\boldsymbol{G}\right)$$

Defferrard et al., NIPS 2016, Kipf et al., ICLR 2017

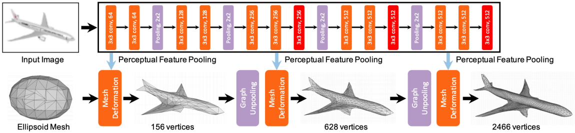
Fixed Topology Mesh Generation with Spectral GNNs

$$\mathbf{F}'(x) = \xi\left(\sum_{y \in \mathcal{N}(x)} T_1(\boldsymbol{\Delta})\mathbf{F}(y)\boldsymbol{G}\right)$$

- ☺ Small number of parameters and easy to optimize
- ☺ Connectivity of the graph explicitly encoded throught the Graph Laplacian ⇒ same transformation applied to corresponding points
- ☹ Reduced expressivity: One parameter per hop ⇒ Isotropic Kernels

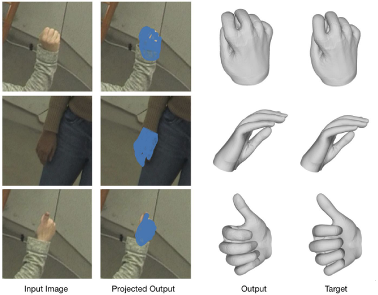
Defferrard et al., NIPS 2016, Kipf et al., ICLR 2017

Fixed Topology Mesh Generation with Spectral GNNs



Pixel2Mesh
Wang et al., ECCV 2018

COMA
Ranjan et al., ECCV 2018



3d Hand Recovery
Kulon et al., BMVC 2019

3D Body Recovery
Kolotouros et al., CVPR 2019

Graph Neural Networks: Attention-based Kernels

- To allow for anisotropy without losing permutation invariance: filters are based on an attention-like mechanism
- Replace the Laplacian Polynomial with learnable weights $w(\boldsymbol{x}, \boldsymbol{y}) \Rightarrow$ each neighbour sends a different message to the central node

$$\boldsymbol{F'}(x) = \xi\left(\sum_{y \in \mathcal{N}(x)} w(\boldsymbol{x}, \boldsymbol{y})\ \boldsymbol{F}(y)\boldsymbol{G}\right),$$

and by allowing multiple kernels $\boldsymbol{G}$:

$$\boldsymbol{F'}(x) = \xi\left(\sum_{k=1}^{K}\Big(\sum_{y \in \mathcal{N}(x)} w_k(\boldsymbol{x}, \boldsymbol{y})\ \boldsymbol{F}(y)\Big)\boldsymbol{G}_k\right)$$

Graph Neural Networks: Attention-based Kernels

- To allow for anisotropy without losing permutation invariance: filters are based on an attention-like mechanism
- Replace the Laplacian Polynomial with learnable weights $w(\boldsymbol{x}, \boldsymbol{y}) \Rightarrow$ each neighbour sends a different message to the central node

$$\boldsymbol{F'}(x) = \xi \left(\sum_{y \in \mathcal{N}(x)} w(\boldsymbol{x}, \boldsymbol{y}) \ \boldsymbol{F}(y) \boldsymbol{G} \right) \,,$$

and by allowing multiple kernels $\boldsymbol{G}$:

$$\boldsymbol{F'}(x) = \xi \left(\sum_{k=1}^{K} \Big(\sum_{y \in \mathcal{N}(x)} w_k(\boldsymbol{x}, \boldsymbol{y}) \ \boldsymbol{F}(y) \Big) \boldsymbol{G}_k \right)$$

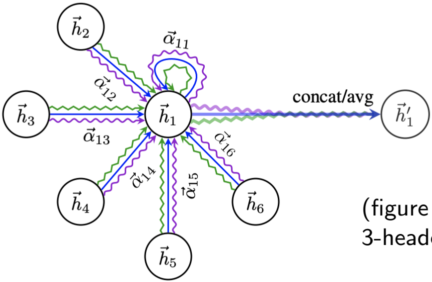
- This is equivalent to performing a soft-mapping (sometimes called patch-operator) between neighbours y and kernels $\boldsymbol{G}_k$, i.e:

$$\boldsymbol{\mathcal{D}}_k(x)(\boldsymbol{F}) = \sum_{y \in \mathcal{N}(x)} w_k(x, y) \boldsymbol{F}(y)$$

Monti et al., CVPR 2017, Verma et al., CVPR 2018, Veličković et al., ICLR 2018

Fixed Topology Mesh Generation with Attention-based GNNs

$$\boldsymbol{F'}(x) = \xi\left(\sum_{k=1}^{K}\Big(\sum_{y\in\mathcal{N}(x)} w_k(\boldsymbol{x},\boldsymbol{y})\ \boldsymbol{F}(y)\Big)\,\boldsymbol{G}_k\right)$$

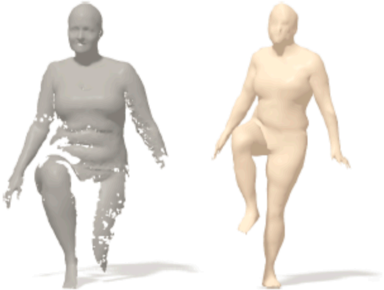


(figure by Petar Veličković:
3-headed graph attention mechanism)

- ☺ Anisotropic Kernels
- ☹ Attention weights are functions of the signal ⇒ No explicit encoding of the connectivity
- ☹ Soft mapping ⇒ Larger number of parameters, can be harder to optimize

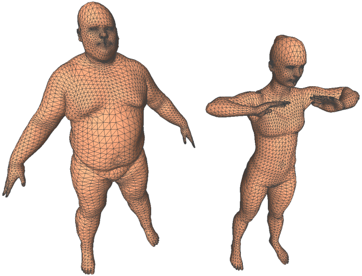
Monti et al., CVPR 2017, Verma et al., CVPR 2018, Veličković et al., ICLR 2018

Fixed Topology Mesh Generation with Attention-based GNNs



Shape Completion with Mesh VAE
Litany et al., CVPR 2018

How to benefit from the advantages of both?



- Spectral methods: Connectivity modelled through the graph Laplacian

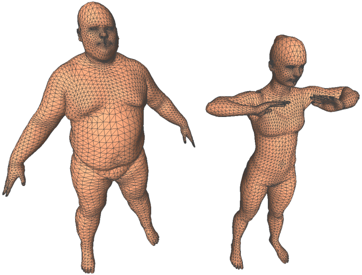
$$\mathbf{F}' = \xi \left(\sum_{r=0}^{R} T_r(\boldsymbol{\Delta}) \mathbf{F}\, \boldsymbol{G}_r \right)$$

☺ Small number of parameters

☺ Different signal values on the same node always undergo the same transformation.

☹ Isotropic Kernels

How to benefit from the advantages of both?



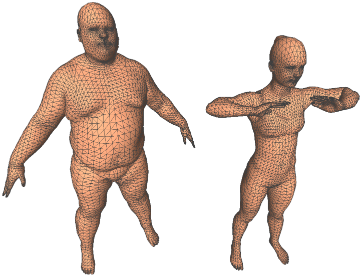
- Attention-based:

$$\boldsymbol{F'}(x) = \xi\left(\sum_{k=1}^{K}\Big(\sum_{y\in\mathcal{N}(x)} w_k(\boldsymbol{x},\boldsymbol{y})\ \boldsymbol{F}(y)\Big)\boldsymbol{G}_k\right)$$

☺ Anisotropic Kernels

☹ Connectivity not explicitly modelled: Different signals values on the same node undergo different transformations

How to benefit from the advantages of both?



- Anisotropic Kernels ⇒ different parameter per neighbour similar to attention-based GNNs
- Small number of parameters and easy-to-optimise ⇒ "Hard" assignments between nodes and parameters. **Attention weights should be either 0 or 1**
- Explicitly encode the connectivity of the graph (fixed topology prior) ⇒ Binary attention weights should depend only on the connectivity

Ordering-Based Graph Convolutions

- **Solution: locally order the vertices**!

Ordering-Based Graph Convolutions

- **Solution: locally order the vertices**!
- Break the permutation invariant constraint that governs all GNNs.
- For kernels equal to the maximum number of neighbours $K = max(|\mathcal{N}(x)|)$:

$$\boldsymbol{F'}(x) = \xi\left(\sum\nolimits_{k=1}^{|\mathcal{N}(x)|} \boldsymbol{F}(x_k)\boldsymbol{G}_k\right).$$

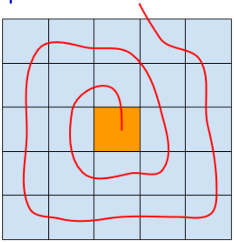
where $\mathcal{N}(x) = \{x_1, \ldots, x_{|\mathcal{N}(x)|}\}$ the neighbourhood of x (inc. x) ordered in some fixed way.

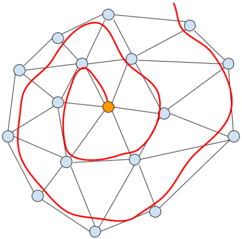
- As a Patch Operator: $\boldsymbol{\mathcal{D}}_k(x)(\boldsymbol{F}) = \boldsymbol{F}(x_k)$
- The above formulation is equivalent with traditional convolution, after choosing a consistent ordering.

Bouritsas*, Bokhnyak* et al., ICCV 2019, ICLRW 2019

How to define the local ordering: Spiral Convolutions

- Consistent ordering across different vertices of the graph via a spiral scan
- Spiral scan:





- Uniquely defined after choosing the starting point and the direction

The ordering needs to remain fixed

Lim et al., ECCVW 2018, Bouritsas*, Bokhnyak* et al., ICCV 2019, ICLRW 2019

Fixed Topology Mesh Generation with Ordering-Based GNNs

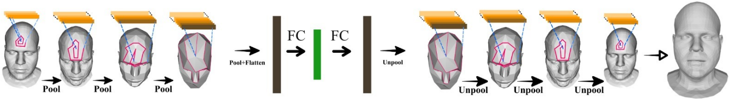
$$\boldsymbol{F'}(x) = \xi\left(\sum_{k=1}^{|\mathcal{N}(x)|} \boldsymbol{F}(x_k)\boldsymbol{G}_k\right)$$

- ☺ Anisotropic Kernels
- ☺ Lightweight, fast & easier to optimise
- ☺ Connectivity and geometry aware
- ☺ Similar to traditional convolutions ⇒ practices for traditional CNNs can be directly transferred (e.g. dilated convolutions)
- ☹ Ordering needs to be engineered

Bouritsas*, Bokhnyak* et al., ICCV 2019, ICLRW 2019

Neural3DMM: Representation Learning for 3D meshes

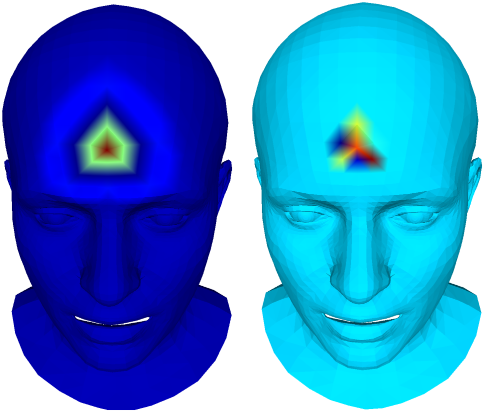
- Autoencoder architecture
- Spiral Convolutions
- Hierarchical structure



Bouritsas*, Bokhnyak* et al., ICCV 2019, ICLRW 2019

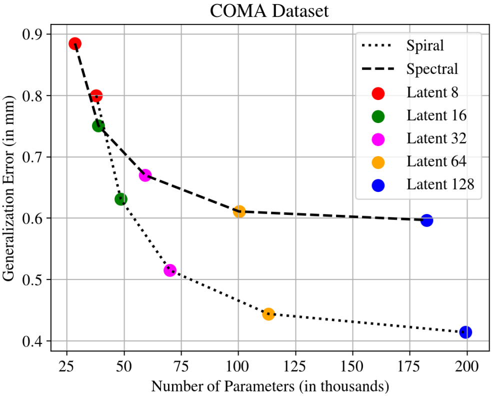
Ordering-based vs Spectral GNNs

- Output of the operator at each vertex (delta function used as input)



Ordering-based vs other GNNs

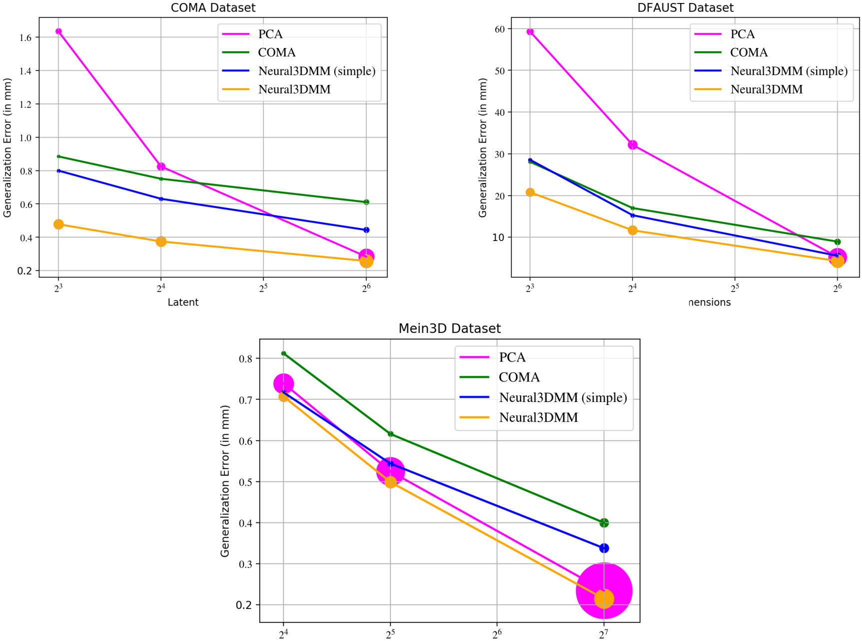
- **Ordering-based vs Spectral GNNs**



- **Ordering-based vs Attention-based GNNs**

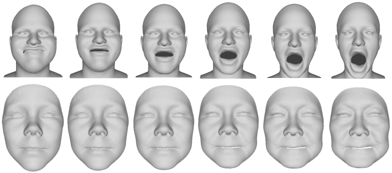
	Monti et al.+ K=9	Monti et al. + K=25	Ours + K=9
error	0.48	0.395	**0.387**
params	401K	940K	**400K**

Fixed Topology Mesh Generation: GNNs vs Statistcal Shape Modelling

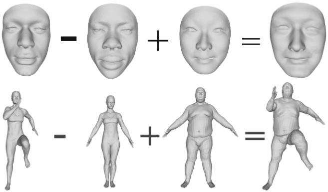


Vector space Arithmetics

- Interpolation



- Analogies



3D Face Synthesis: Wasserstein GAN with GP

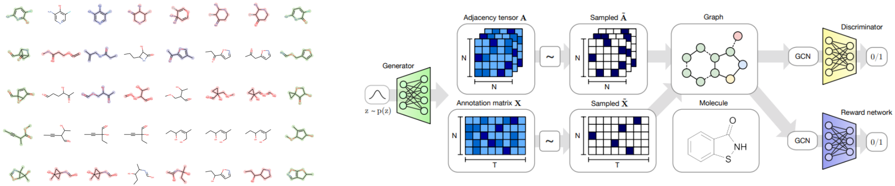
Arbitrary topology 3D shape generation: Relatively unexplored

Range scan to 3D mesh (link prediction on a fully connected graph ⇒ only up to 100 verts)

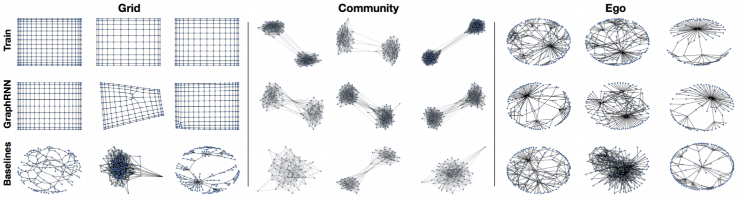
Fixed Topology and adaptive face splitting (Zero genus shape generation)

Dai and Nießner, CVPR 2019, Smith et al., ICML 2019

Can we draw insipration from methods on arbitrary Graphs?



Molecule generation



Topology Generation

Simonovsky and Komodakis, ICANN 2019, De Cao and Kipf, ICMLW 2018, You et al., ICML 2018

Conclusions

- 3D shape generation still has a long way to go
- Only recently the community invented the right tools to work directly on non-euclidean domains

Conclusions

- 3D shape generation still has a long way to go
- Only recently the community invented the right tools to work directly on non-euclidean domains
- Fixed Topology mesh generation:
  - Enforcing an ordering can be beneficial and drastically reduces the number of parameters
  - Still the ordering is engineered
  - Can we learn it?
  - Can we enforce orderings on arbitrary underlying graphs?

Conclusions

- 3D shape generation still has a long way to go
- Only recently the community invented the right tools to work directly on non-euclidean domains
- Fixed Topology mesh generation:
  - Enforcing an ordering can be beneficial and drastically reduces the number of parameters
  - Still the ordering is engineered
  - Can we learn it?
  - Can we enforce orderings on arbitrary underlying graphs?
- Arbitrary Topology mesh generation:
  - Still no consensus on how to approach the problem
  - Graph generation? Implicit Surfaces? Spectral Domain? Gaussian Processes? ...

Conclusions

- 3D shape generation still has a long way to go
- Only recently the community invented the right tools to work directly on non-euclidean domains
- Fixed Topology mesh generation:
  - Enforcing an ordering can be beneficial and drastically reduces the number of parameters
  - Still the ordering is engineered
  - Can we learn it?
  - Can we enforce orderings on arbitrary underlying graphs?
- Arbitrary Topology mesh generation:
  - Still no consensus on how to approach the problem
  - Graph generation? Implicit Surfaces? Spectral Domain? Gaussian Processes? ...