OccFacto: Controllable Part-Based Mesh Generation with Occupancy Networks

Aniketh Iyengar[a], William Pan[b,c]

[a] Department of Computer Science, Stanford University; [b] Department of Mechanical Engineering, Stanford University; [c] Department of Electrical Engineering, Stanford University

Stanford University

Significance.

We present an novel model that:
(1) takes an input of part-segmented styles of 3D objects
(2) learns an implicit function that outputs occupancy to represent 3D objects as meshes
(3) enables generation of coherent and plausible 3D objects with part-based control

Applications

Implications for optimizing generation 3D objects for product design, mechanical engineering, and manufacturing sector

Background.

3D object representations.

Point Clouds

- **Pros:** simple data representation of 3D points
- **Cons:** lack boundary representations

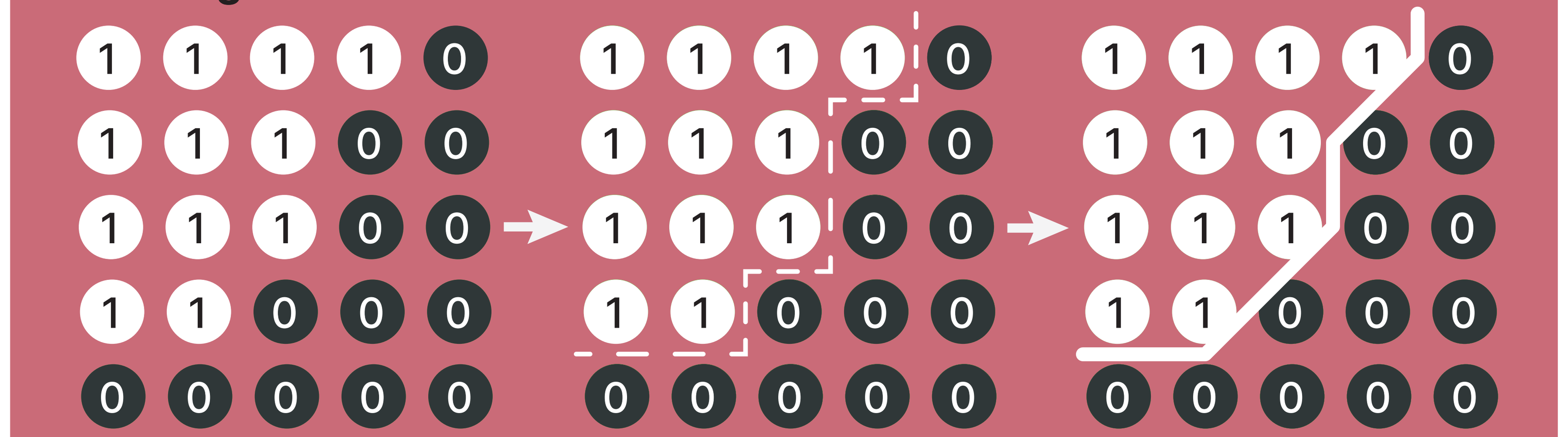
How can we turn easy-to-sample point clouds to fully-defined meshes?

Meshes

- **Pros:** fully represent a shape's geometry
- **Cons:** data representation is a graph of edges and faces

Occupancy Networks.

Learning Meshes from Point Clouds

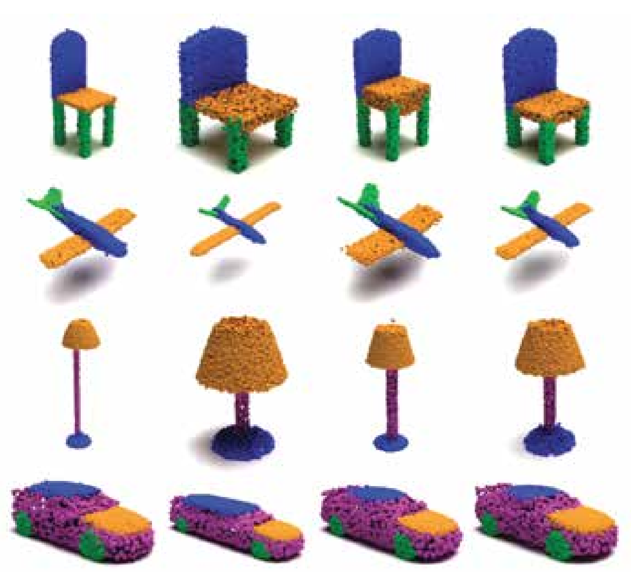


Zeros represent **outside** of object while ones represent **inside**

We can have a function learn a **decision boundary**

Apply **Marching Cubes algorithm**[1] to generate meshes

Part-Based Shape Generation.



DiffFacto[2]: Factorized Representations with Cross Difussion

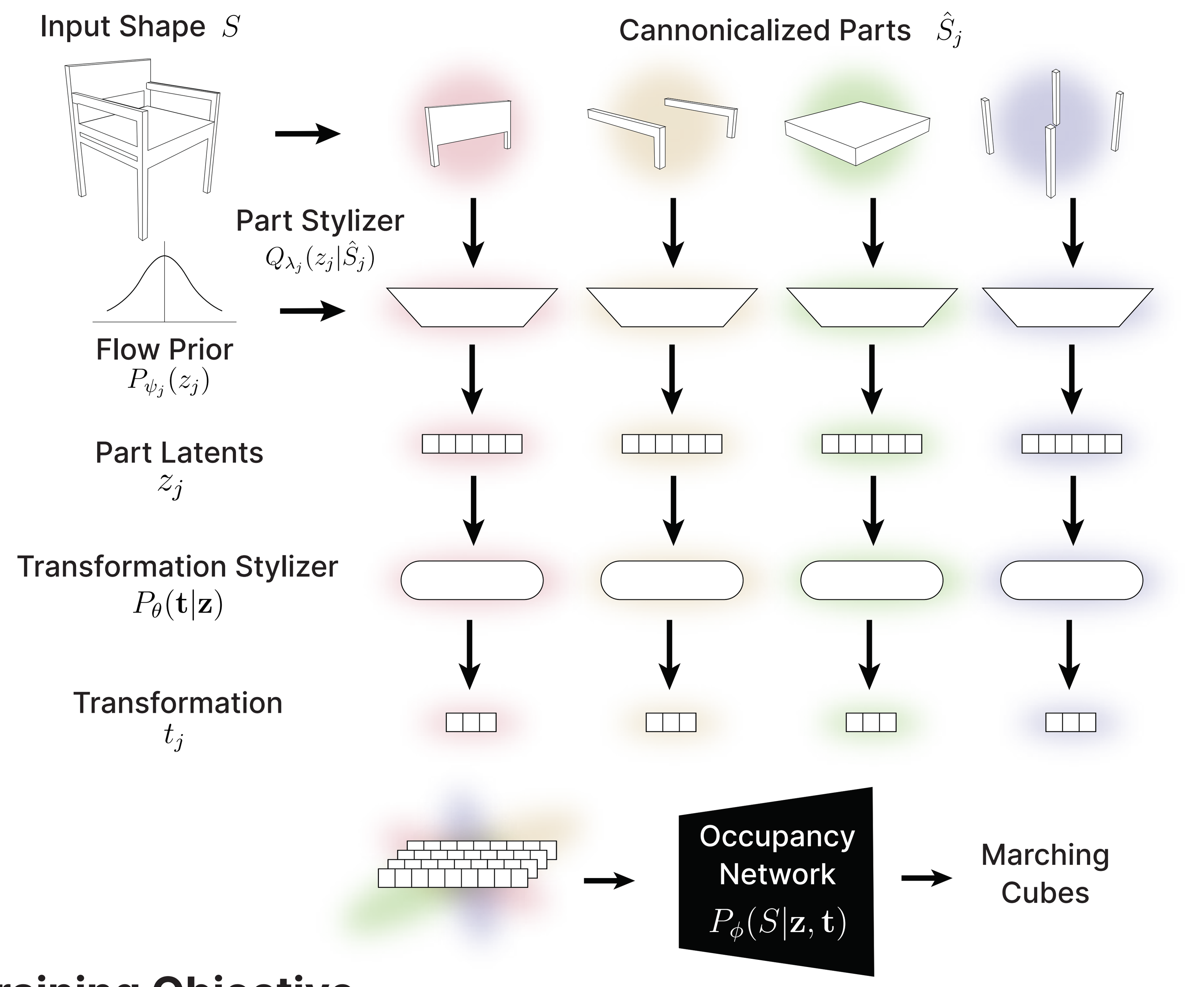
Part stylizer variational encoder (PointNet[3]), flow network prior, transformation sampler self-attention transformer, cross diffusion point cloud network

SPAGHETTI[4]: Editing Implicit Shapes Through Part Aware Generation

- Transformer cross-attention only occupancy network decoder
- Inspired by DeepSDF[5] (transformer with signed distance fields)
- GMM part decomposition

Methods.

Dataset. **ShapeNet[6] v1 part-segmented chairs (3053 train, 704 test)**
Inputs: Shapes encoded by 2048 points from point cloud
Occupancy: Trained via 2048 points outside randomly sampled in bounding cube



Training Objective.

$$\mathbb{E}_S[\log P_{\psi,\theta,\phi}(S)] = \mathbb{E}_S\left[\log \int\int P_\phi(S|\mathbf{z},\mathbf{t})P_{\psi,\theta}(\mathbf{z},\mathbf{t})\,d\mathbf{z}\,d\mathbf{t}\right]$$

$$\geq \mathbb{E}_{S,\mathbf{z}}[\log P_{\lambda||\phi||_2^2}(S|\mathbf{z},\mathbf{t}) + \sum_{j=1}^{m}\log\frac{P_{\psi_j}(z_j)}{Q_{\lambda_j}(z_j|\hat{S}_j)} + \log P_\theta(\mathbf{t}|\mathbf{z})] = \mathbb{E}[\mathcal{L}_{recon} + \mathcal{L}_z + \mathcal{L}_t]$$

- Part Stylizer and Transformation Sampler Evidence Lower Bound (ELBO) already maximized through training of DiffFacto. Utilize pretrained weigthts.
- Maximize ELBO via optimizing reconstruction loss with binary cross entropy

$$\mathcal{L}_{recon} = -\frac{1}{|X|}\sum_{(x,y,o)\in X}\mathrm{BCE}(o, f_\phi(x,y)) + \epsilon||\phi||_2^2$$

Optimization.

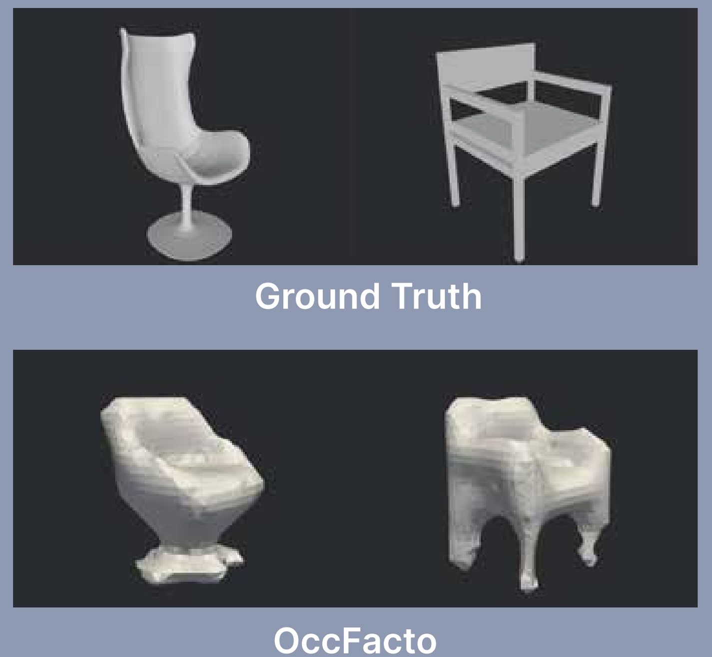
- **Linear Warm-Up** over 1000 iterations to 5e-4
- **Exponential Decay** (beta1 = 0.9, beta2 = 0.999, epsilon = 1e-8) every 120 epochs
- λ = 0.01 **L2 Regularization** and 0.1 dropout multi-head attention **transformer layers and feed forward** to prevent overfitting caused by frozen weights in encoder

Future Work.

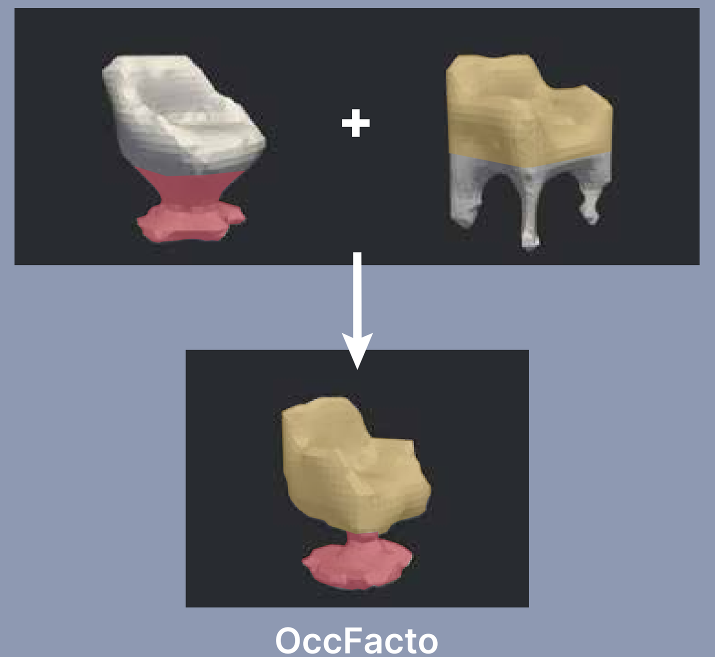
1. **Limited Compute Related:** Greater hyper-parameter tuning, Increasing point samples, Full-pipeline training (Unfreezing DiffFacto).
2. **Further Steps:** Alternative Occupancy Architectures, Text Conditioned Generation.

Results.

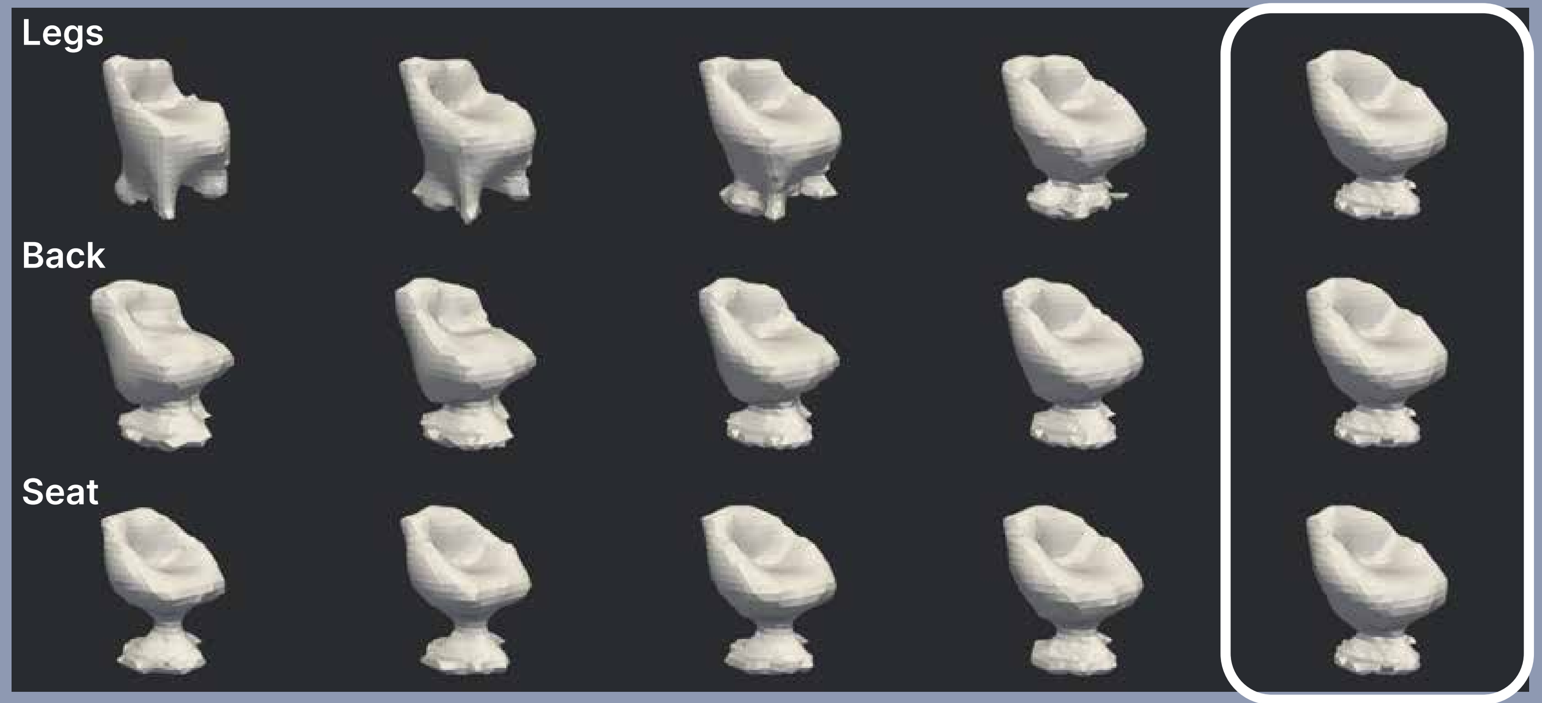
Shape Reconstruction.



Style Changes.



Part Style Interpolation



Metrics

	Binary Accuracy	Volumetric IOU	Chamfer (L1)
OccFacto	**89.04%**	0.6471	**0.0776**
OccNet[7]	N/A	**0.778**	0.079
Linear	64.93%	0.487	0.112

Citations.
[1] Lorensen, W. E., & Cline, H. E. (1987). ACM SIGGRAPH Computer Graphics; [2] Nakayama, K., et al. (2023). (arXiv:2305.01921). arXiv.; [3] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). (arXiv:1612.00593).; [4] Hertz, A. et al. (2022). (arXiv:2201.13168).; [5] Park, J. J. et al. (2019). (arXiv:1901.05103). arXiv; [6] Chang, A. X. et al. (2015). (arXiv:1512.03012).; [7] Mescheder, L. et al. (2019). (arXiv:1812.03828).

Acknowledgements. AI and WP would like to thank Mikaela (Mika) Uy and George Nakayama from the Guibas Lab for their assistance with the project; Stefano Ermon and the CS 236 Teaching Assistants; and friends and family.