Combinatorial Construction via Sequential Assembly

Jungtaek Kim, Ph.D.
jtkim@postech.ac.kr

POSTECH
Pohang 37673, Republic of Korea
https://jungtaek.github.io

April 14, 2022

Table of Contents

Introduction

Combinatorial Construction

Combinatorial 3D Shape Generation via Sequential Assembly

Building LEGO Using Deep Generative Models of Graphs

Brick-by-Brick: Combinatorial Construction with Deep Reinforcement Learning

Discussion & Conclusion

Introduction

Today's Talk

- I will introduce three papers on combinatorial construction today.
- These papers solve a combinatorial construction problem that assembles unit primitives (i.e., 2×4 LEGO bricks) under a fixed rule.

Method	Representation	Target Info.	Approach
Kim et al. [2020]	Set	Exact target volume	Bayesian optimization
Thompson et al. [2020]	Graph	One-hot class info.	Graph generative model
Chung et al. [2021]	Graph	Images	Reinforcement learning

[Kim et al., 2020] **J. Kim**, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. In Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng), Virtual, 2020.

[Thompson et al., 2020] R. Thompson, G. Elahe, T. DeVries, and G. W. Taylor. Building LEGO using deep generative models of graphs. In Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng), Virtual, 2020.

[Chung et al., 2021] H. Chung*, **J. Kim***, B. Knyazev, J. Lee, G. W. Taylor, J. Park, and M. Cho. Brick-by-Brick: Combinatorial construction with deep reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS), volume 34, pages 5745–5757, Virtual, 2021.

More Detailed Comparisons

Table 1: Analysis of recent studies in terms of state representation, supervision, conditioning, target objects, and action validation.

Method	State	Supervision	Conditioning	Target	Action Validation
Hamrick et al. [2018]	Image	Task-dependent	N/A	2D	Direct
Bapst et al. [2019]	Object/Image	Task-dependent	Object and/or image	2D	Direct
Kim et al. [2020]	Set	Overlap	Exact target volume	3D	Sampling
Thompson et al. [2020]	Graph	Step-wise CE	One-hot class info.	3D	Direct
Chung et al. [2021]	Graph	IoU	Image or set of images	3D	Pretrained

[Hamrick et al., 2018] J. B. Hamrick, K. R. Allen, V. Bapst, T. Zhu, K. R. McKee, J. B. Tenenbaum, and P. W. Battaglia. Relational inductive bias for physical construction in humans and machines. In Proceedings of the Annual Conference of the Cognitive Science Society (CogSci), pages 1773–1778, Madison, Wisconsin, USA, 2018.

[Bapst et al., 2019] V. Bapst, A. Sanchez-Gonzalez, C. Doersch, K. Stachenfeld, P. Kohli, P. Battaglia, and J. Hamrick. Structured agents for physical construction. In Proceedings of the International Conference on Machine Learning (ICML), pages 464–474, Long Beach, California, USA, 2019.

[Kim et al., 2020] **J. Kim**, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. In Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng), Virtual, 2020.

[Thompson et al., 2020] R. Thompson, G. Elahe, T. DeVries, and G. W. Taylor. Building LEGO using deep generative models of graphs. In Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng), Virtual, 2020.

[Chung et al., 2021] H. Chung*, **J. Kim***, B. Knyazev, J. Lee, G. W. Taylor, J. Park, and M. Cho. Brick-by-Brick: Combinatorial construction with deep reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS), volume 34, pages 5745–5757, Virtual, 2021.

Introduction

- Combinatorial construction via *sequential assembly* mimics a human assembly process, by allocating a budget of primitives given.
- We introduce a sequential problem by utilizing an approach to solving *combinatorial 3D shape generation*.

Combinatorial Construction

Combinatorial Construction

Figure 1: A 2×4 brick.

- A 2×4 brick is used as a unit primitive, which has eight studs and their fit cavities.
- Instead of employing other 3D representations such as point clouds, triangular meshes, and voxels, we create a sequence of unit primitives.
- Interestingly, possible positions to assemble a brick consistently grow.
- With only *six* 2×4 bricks, *915,103,765* combinations possible exist [Eilers, 2016].

[Eilers, 2016] S. Eilers. The LEGO counting problem. The American Mathematical Monthly, 123(5):415–426, 2016.

Combinatorial Construction: Assumptions

- Every connection types must follow a fixed rule.

Figure 2: Example of available offsets.

- No bricks mutually overlap.

Figure 3: Taken from [Yang et al., 2019].

- If two bricks are connected, the combination of two bricks is considered that they are soldered together and cannot be broken.

[Yang et al., 2019] B. Yang et al. Learning object bounding boxes for 3D instance segmentation on point clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 6740–6749, 2019.

Combinatorial Construction: Action Space

Figure 4: Successive action space.

Combinatorial 3D Shape Generation via Sequential Assembly

Sequential Assembly with Unit Primitives

- Such 2×4 LEGO bricks make our problem more combinatorial and more complex, compared to other primitives.

(a) Target shape (b) 1×1-sized primitives (c) 2×4-sized primitives

Figure 5: Assembly of a target shape with 1×1-sized primitives and 2×4-sized primitives.

Combinatorial 3D Shape Generation

- To determine the position of the next primitive, we define two evaluation functions regarding *occupiability* and *stability*.
- Occupiability encourages us to follow a target shape and stability helps to create a physically-stable combination.
- We determine the position of the next primitive via *Bayesian optimization*.
- To avoid a suboptimal sequence, our framework includes a *rollback step*.

(a) Placing an object (b) Applying forces (c) Measuring stability

Figure 6: Stability simulation with `PyBullet`.

Experimental Results

Figure 7: Generated assembling sequence that creates a *car* shape with 118 unit primitives.

Experimental Results

- We apply our framework in optimizing particular explicit functions.

(a) Height (b) Width (c) Depth (d) #Connected studs

Figure 8: Quantitative results on maximizing explicit evaluation functions.

Combinatorial 3D Shape Dataset

- We also introduce a new *combinatorial 3D shape dataset* that consists of 14 classes and 406 instances.

Figure 9: Selected examples from our dataset.

Combinatorial 3D Shape Dataset

- The characteristics of our combinatorial 3D shape dataset are
  1. *combinatorial*: Duplicates of unit primitive is repeatedly connected;
  2. *sequential*: Allowable connections between primitives are sequentially added;
  3. *decomposable*: By the combinatorial property, parts of combination can be sampled if they are valid in terms of the contact and overlap conditions;
  4. *manipulable*: New primitive is addable or the existing primitives are removable.

Building LEGO Using Deep Generative Models of Graphs

Graph Representation

- It represents a connection between two bricks (denoted as i and j) by a displacement:

$$\mathbf{e}_{ij} = \mathbf{x}_i - \mathbf{x}_j, \tag{1}$$

which can be used as an edge attribute.

Deep Generative Model of LEGO Graphs (DGMLG)

- To generate a LEGO construction, the authors propose Deep Generative Model of LEGO Graphs (DGMLG), based on DGMG [Li et al., 2018].
- Combinatorial 3D shape dataset [Kim et al., 2020] is used to train the proposed models.
- DGMLG-Re, which is a variant of DGMLG, prevents selecting any actions that would result in an invalid structure.

[Li et al., 2018] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. W. Battaglia. Learning deep generative models of graphs. arXiv preprint arXiv:1803.03324, 2018.

[Kim et al., 2020] **J. Kim**, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. In Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng), Virtual, 2020.

Experimental Results

	FD ↓	GIN Acc ↑	P ↑	R ↑	D ↑	C ↑	KD ↓	% novel ↑	% valid ↑
BO-SI [9]	345	24.5	0.47	0.70	0.23	0.078	5297	93.5	**100**
BO-CC [9]	407	11.9	0.42	0.59	0.26	0.051	8901	**100**	**100**
DGMLG	**150**	**60.5**	**0.62**	**0.92**	**0.48**	**0.23**	**2054**	89.5	25
DGMLG-Re	334	50.5	0.59	0.88	0.44	0.21	2.4e4	87	**100**

Figure 10: Performance evaluation of DGMLG vs. the Bayesian optimization baselines on the proposed GIN-based metrics.

Brick-by-Brick: Combinatorial Construction with Deep Reinforcement Learning

Overview

- It sequentially assembles unit primitives (i.e., LEGO bricks), given only incomplete target information (i.e., a 2D image or multiple views of a target object).
- It requires a comprehensive understanding of incomplete target information and long-term planning to append each brick efficiently.

Overview

- We devise a reinforcement learning (RL) approach along with the absence of sequence-level supervision.
- We express a brick combination as graph representation, where node and edge correspond to a single brick and a connection between two bricks, respectively.
- In this domain, however, we struggle to handle both an indefinite action space and the existence of many invalid actions when applying RL.
- To resolve the aforementioned issues, we adopt an action validity prediction network that filters out invalid actions to an actor-critic network.

Combinatorial Construction: Overall Scenario

Figure 11: Training and test episodes of combinatorial construction.

Brick-by-Brick (B^3)

- Suppose that *target information* $\mathcal{T}$, i.e., a single binary image or a set of three binary images from different views of a target object, is given as partial information.
- Represent each t-th *state* s_t of the MDP as a tuple of a directed graph G_t composed of t bricks and target information $\mathcal{T}$, i.e., $s_t = (G_t, \mathcal{T})$.
- With t bricks assembled, define an *action* $a_t = (a_t^{\mathrm{piv}}, a_t^{\mathrm{off}})$ where a_t^{piv} is to select a pivot brick and a_t^{off} is to select an offset with respect to the pivot brick.
- Transform the combination of currently assembled bricks into the occupancy of the voxels and measure the overlap between them as a *reward function*:

$$\Delta\mathrm{IoU}(\mathbf{C}_t, \mathbf{T}) = \frac{\mathrm{vol}(\mathbf{C}_t \cap \mathbf{T})}{\mathrm{vol}(\mathbf{C}_t \cup \mathbf{T})} - \frac{\mathrm{vol}(\mathbf{C}_{t-1} \cap \mathbf{T})}{\mathrm{vol}(\mathbf{C}_{t-1} \cup \mathbf{T})}, \tag{2}$$

where $\mathbf{C}_t$, $\mathbf{C}_{t-1}$, and $\mathbf{T}$ are the occupied voxels at timestep t, timestep $t-1$, and a desired target, respectively. Note that $\mathrm{vol}(\cdot)$ is a function that measures a volume.

Brick-by-Brick (B^3)

Figure 12: An overview of our proposed method B^3. The red brick in both a_t^{piv} and offset a_t^{off} indicates the chosen brick.

- We use a respective graph neural network for a_t^{piv} and a_t^{off}, inspired by Battaglia et al. [2018].

[Battaglia et al., 2018] P. W. Battaglia et al. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261, 2018.

Action Validity Prediction Network

- An action validity prediction network predicts an invalid action using a surrogate for confirming the validity of a given action.
- It trains a graph neural network, of which the head corresponds to the degree of validity.
- It capables of pre-training the graph neural network with the ground-truth validity of actions, which is obtained by randomly-assembled objects.
- It applies the network in training an actor-critic network, without re-training.

Experimental Results: MNIST Construction

Figure 13: Qualitative results on MNIST construction.

Experimental Results: Randomly-Assembled Object Construction

(a) Target images (b) Constructed object from three viewpoints

Figure 14: Qualitative results on randomly-assembled object construction.

Experimental Results: ModelNet Construction

(a) Airplane 1 (b) Airplane 2

(c) Monitor (d) Table

Figure 15: Qualitative results on ModelNet construction.

Experimental Results: Episode Return Curves

(a) MNIST (b) Randomly-Assembled (c) ModelNet

Figure 16: Episode return curves vs. timesteps in different setups. The curves measured by training and test episodes are reported by repeating 3 times with different seeds.

Experimental Results: Analysis on Action Validity Prediction Network

(a) Pivot ROC (b) Pivot PR (c) Offset ROC (d) Offset PR

Figure 17: ROC and PR curves for action validity prediction networks.

Discussion & Conclusion

Discussion

- We open a new line of research on a sequential and combinatorial construction problem.
- The methods covered in this talk are limited to use a 2×4 LEGO brick as a unique unit primitive.
- We need to investigate more realistic rewards such as stability and feasibility, because our method finds a way to construct a 3D object.
- We need to suggest better graph representation to reflect the state of brick combination.
- We would like to create a large-scale graph dataset, which contains randomly-assembled objects with diverse types of unit primitives, e.g., 2×4, 2×2, and 1×2 bricks.

Conclusion

- We propose the problem formulation of sequential assembly for a combinatorial construction problem.
- This line of research shows that it can successfully generate a 3D object in a combinatorial manner.
- We investigate two representations, i.e., sets and graphs, and three approaches, i.e., Bayesian optimization, deep graph generative models, and deep reinforcement learning.
- Also, we create a new dataset for combinatorial 3D models, which allows us to generate 3D shapes sequentially.

Any Questions?

References I

V. Bapst, A. Sanchez-Gonzalez, C. Doersch, K. Stachenfeld, P. Kohli, P. Battaglia, and J. Hamrick. Structured agents for physical construction. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 464–474, Long Beach, California, USA, 2019.

P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

H. Chung, J. Kim, B. Knyazev, J. Lee, G. W. Taylor, J. Park, and M. Cho. Brick-by-Brick: Combinatorial construction with deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 5745–5757, Virtual, 2021.

S. Eilers. The LEGO counting problem. *The American Mathematical Monthly*, 123(5):415–426, 2016.

J. B. Hamrick, K. R. Allen, V. Bapst, T. Zhu, K. R. McKee, J. B. Tenenbaum, and P. W. Battaglia. Relational inductive bias for physical construction in humans and machines. In *Proceedings of the Annual Conference of the Cognitive Science Society (CogSci)*, pages 1773–1778, Madison, Wisconsin, USA, 2018.

J. Kim, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. In *Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng)*, Virtual, 2020.

Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. W. Battaglia. Learning deep generative models of graphs. *arXiv preprint arXiv:1803.03324*, 2018.

R. Thompson, G. Elahe, T. DeVries, and G. W. Taylor. Building LEGO using deep generative models of graphs. In *Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng)*, Virtual, 2020.

B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, and N. Trigoni. Learning object bounding boxes for 3D instance segmentation on point clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 6740–6749, 2019.