GCM

EPFL

Student Project Proposal

C-shells Generation

Quentin BECKER – <quentin.becker@epfl.ch> – BC 346
Geometric Computing Laboratory

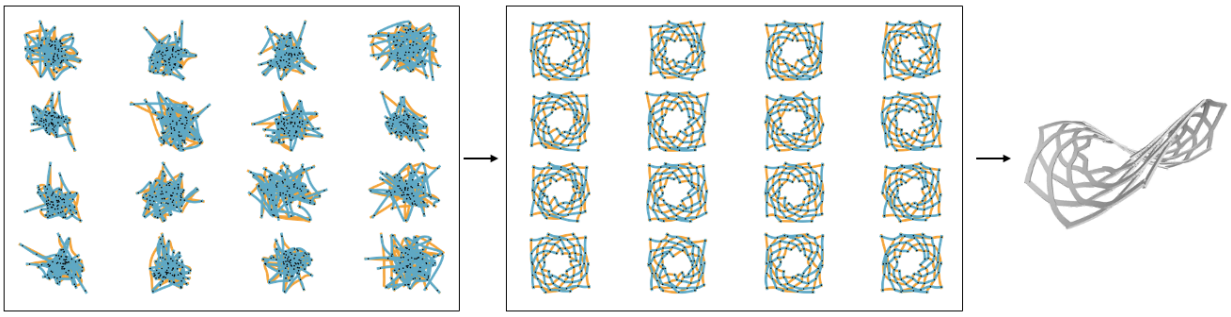

Figure 1: The generation process. *Left*: the generative model is randomly initialized and is likely to produce self-intersecting flat linkages. *Center*: after few iterations, the generative model has learnt to disentangle the curves linkage. *Right*: once the curves are drawn, the rod linkage can be simulated and deployed.

Description

C-shells are deployable structures made of elastic rods that are connected using rotational joints. C-shells extend the existing X-shells [4] by allowing beams to have non-zero curvature in their rest state. The rod linkage can be designed such that it has a stress-free and flat assembly state, making the assembly process easier compared to X-shells.

In practice, designing a C-shell involves several steps. First, the designer must specify the linkage topology i.e., the beam layout. Second, curved beams are drawn while respecting the prescribed topology. Third, the rod linkage is simulated and deployed. The designer can iterate on the first and second steps until the deployed shape satisfies their taste. So far, only a few parametric families serve creating the models and this project aims at extending the range of simulated C-shells by means of generative models.

Generative models have shown impressive results in faithfully reproducing high quality images of celebrities, bedrooms, or other diverse objects [2]. This class of models have been extensively studied since the introduction of Variational Auto-Encoders (VAE) [3] and Generative Adversarial Networks (GAN) [1]. Still, such training methods usually assume the existence of a large volume of data. Unfortunately, such a dataset does not exist in the case of C-Shells generation.

The goal of this project is to adapt the generative model training framework, so that it can be trained using a very limited amount of data. In particular, the student will be confronted to the following topics:

- Curve linkage representation. This first necessary step determines the how the generative model will "perceive" the curves linkage. For instance, tuning the representation can bias the model towards producing symmetric designs.
- Differentiable performance measure. The generative model will not have access to examples. First order information that is necessary to train such models must be found in some other

way *e.g.*, using a performance measure. Such measure would both provide a performance score as well as a direction of improvement to the generator.

- Diversity. Using a performance measure to train the generative model is likely to lead to mode collapse *i.e.*, when the generator only outputs one curves linkage. Enforcing diversity is necessary.

Prerequisites

Good knowledge of linear algebra and machine learning is required; familiarity with generative models is preferred. Good coding skills in Python and in PyTorch is expected.

Remarks

This project is intended for Master's degree students.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[2] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020.

[3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[4] Julian Panetta, MINA Konaković-Luković, Florin Isvoranu, Etienne Bouleau, and Mark Pauly. X-shells: A new class of deployable beam structures. *ACM Transactions on Graphics (TOG)*, 38(4):1–15, 2019.