Supplementary for Learning to Fill the Seam by Vision: Sub-millimeter Peg-in-hole on Unseen Shapes in Real World

Liang Xie[✉]**, Hongxiang Yu, Yinghao Zhao, Haodong Zhang, Zhongxiang Zhou, Minhang Wang, Yue Wang**[✉]**, Rong Xiong**[✉]

State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, Zhejiang, China

1. Introduction

In this document, we provide the details mentioned in the main paper, including the loss function for RL referred in III-B, which we use for the policy training, the peg information (shapes, names, and the corresponding index referred in IV), simulation results of unseen shape generalization, and the results of feature visualisation in IV.

A. Loss Function for RL

The RL agent starts by trial and error to generate a trajectory with random actions a. By increasing exploitation and reducing exploration over time, the agent strives to maximize the expected cumulative rewards:

$$R(\tau) = \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t) \tag{1}$$

$$J(\pi) = \mathbb{E}_{\tau \sim \pi}[R(\tau)] \tag{2}$$

where τ is the episode, T is the episode length, π is the parameters of the policy, γ is the defined discount factor, r is the defined reward function, s_t is the defined system state, and a_t is the defined action. By calculating the derivation of (2), we define the loss function for RL as:

$$\nabla_\pi J(\pi) = \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{T} \nabla_\pi \log \pi(a_t|s_t) R(\tau)\right] \tag{3}$$

We have represented our policy with neural network π which allows us to calculate $\nabla_\pi \log \pi(a_t|s_t)$, and we are able to run the policy in the environment to collect the trajectory dataset, then we can compute the policy gradient and take an update step.

B. Peg Shapes

As shown in Fig. 1, The peg shapes are selected elaborately from daily insert tasks which range from simple to complex, from concave to convex, and from square-corner to round-corner. We train our model on the seen shapes and evaluate it on the unseen shapes as described in Tab. D. The tolerance of the peg-hole pair is 0.5mm$\sim$0.7mm with around 0.2mm 3D print error.

C. Unseen Shape Generalization

To verify the generalization ability of our proposed method to different peg shapes, we conduct a simulation experiment on the 4 seen peg shapes and the 10 unseen peg shapes within 5mm initial position error. The tolerance of the peg-hole pair is 1mm. The success rate is averaged over 12 inserting trials on each peg shape. As shown in Tab. D, nearly 100% insert accuracy is achieved within 5s for all the seen and unseen peg shapes, which proves the unseen shape generalization of the proposed method.

D. Feature Visualization

To figure out what have been learned inside the SNF, we perform feature visualization to display the outputs of the position module and the orientation module as shown in Fig. 2, where the first column is the segmented images of four peg-seam pairs with different shapes. The second column is the one-hot heatmaps produced by the position module, and the third is the 3-channel feature maps of the orientation module with the segmented image I_s as input. We can find that the brightest point relative to the center point in the heatmaps is consistent with the

Table 1. Generalization to Different Peg Shapes

	Shapes index	Shapes name	Success	Time(s)
seen	1	triangle	12/12	2.4 ± 1.29
	2	square	12/12	3.5 ± 0.69
	3	pentagon	12/12	3.0 ± 0.85
	4	hexagon	12/12	2.6 ± 0.32
unseen	1	diamond	12/12	2.7 ± 2.41
	2	trapezoid	12/12	3.0 ± 0.19
	3	fillet-1	12/12	2.7 ± 0.70
	4	fillet-2	12/12	2.0 ± 0.51
	5	fillet-3	12/12	3.2 ± 0.83
	6	fillet-4	12/12	3.5 ± 1.30
	7	concave	12/12	4.9 ± 1.21
	8	convex-1	12/12	3.7 ± 1.37
	9	convex-2	12/12	3.3 ± 0.84
	10	cross	11/12	4.0 ± 1.29

relative position between the peg and the seam, which indicates that the position module focuses on the relative pose while avoiding to overfit to the geometry features. In the 3-channel feature maps, the orientation module learns to encoder the general features (corner, line features) of the seam for orientation alignment. In general, the modules are insensitive to the shape differences but learn to encode general features to determine the relative pose between the peg and the seam, which explains the the generalization of the proposed SFN model.

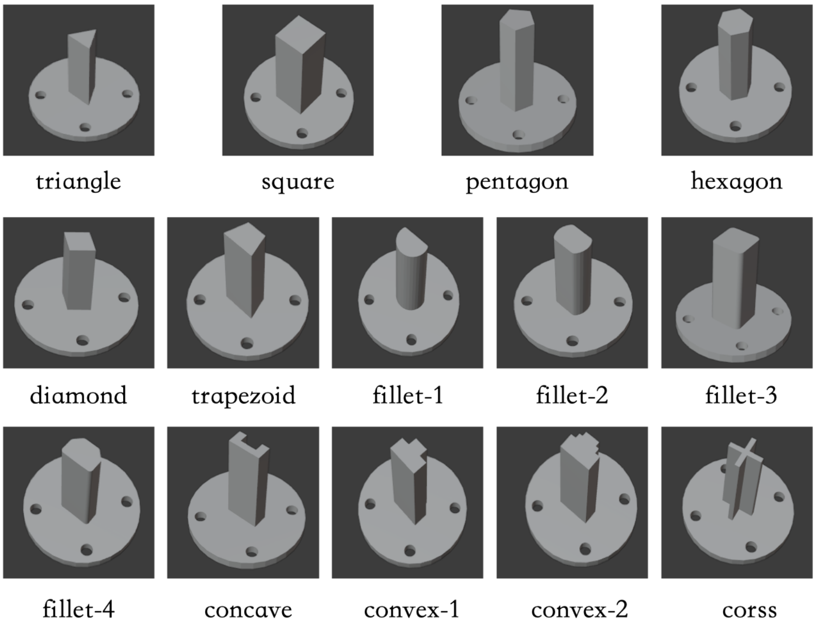

Fig. 1. Peg Shapes

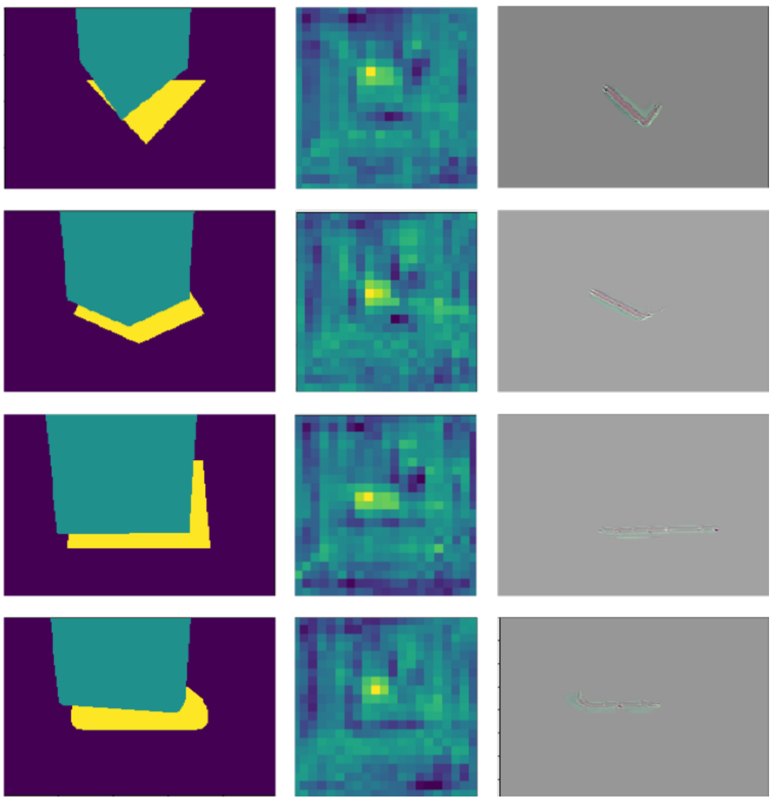

Fig. 2. Feature visualization