Learning Self-prior for Mesh Denoising using Dual Graph Convolutional Networks –Supplementary Document–

Shota Hattori, Tatsuya Yatagawa, Yutaka Ohtake, and Hiromasa Suzuki

School of Engineering, The University of Tokyo
{hattori,tatsy,ohtake,suzuki}@den.t.u-tokyo

A Choice of Input Features

Although we explained the original DIP [29] input a static random code to the neural network, the paper of DIP also reported the better performance of different input features for specific tasks (e.g., they input pixel coordinates for the sake of image completion). Therefore, we investigate what features are appropriate to reproduce positional displacements and facet normals. We compare the denoising performances of several input features and evaluate their effectiveness for mesh denoising.

In this experiment, we consider six feature vectors, which are (r6) the 6D random vector, (r16) the 16D random vector, (p-r) the 3D coordinates of a position (the position of vertex for PosNet, but that of face's centroid for NormNet) concatenated with a 3D random vector, (n-r) the 3D normal vector (of a vertex for PosNet, but of a face for NormNet) concatenated with a 3D random vector, (p-n) the 3D coordinates of a position concatenated with the 3D normal vector, and (p-n-a) the 7D vector comprised of the 3D coordinates of a position, 3D normal vector, and area of a facet. The last one is tested only with NormNet because it uses the area of a facet. Note that the positions and normals included by the feature vectors involve noise because our method only uses noisy input meshes.

The results for PosNet and NormNet are shown in Tabs. A.1 and A.2, respectively. According to Tab. A.1, random vectors (i.e., (r6) and (r16)), rather than those including vertex positions and normals, obtain better reproduction for positional displacements. Intuitively, we may obtain better performances if input features are correlated with positional displacement. However, the coordinates of positions do not correlate with the magnitude of vertex displacements. Neither do the normal vectors, although the orientation of the displacement may somewhat correlate with them. According to these observations, the performances using (p-r), (n-r), and (p-n) may not have been very high. In contrast, it is demonstrated that random vectors obtain the structural patterns of positional displacements due to the nature of neural networks.

On the contrary, the results in Tab. A.2 suggest that the feature vectors involved with positions and normals increase the performance of NormNet. During the experiment, we found that the convergence of training NormNet became

Table A.1. The AADs obtained by DDMP where PosNet is given different input features.

Model	Type	r6	**r16**	p-r	n-r	p-n
sharp	CAD	5.21	**4.90**	5.23	5.40	5.43
twelve	CAD	**1.09**	1.20	1.63	**1.09**	1.63
carter	non-CAD	5.00	**4.95**	7.34	5.36	7.33
grayloc	non-CAD	7.23	7.24	7.10	7.14	**7.09**

Table A.2. The AADs obtained by DDMP where NormNet is given different input features.

Model	Type	r6	r16	p-r	n-r	p-n	**p-n-a**
sharp	CAD	6.39	5.47	5.50	5.32	**4.90**	**4.90**
twelve	CAD	1.32	2.23	1.92	**1.17**	1.34	1.20
carter	non-CAD	4.93	**4.88**	6.13	5.47	**4.93**	4.95
grayloc	non-CAD	11.74	11.84	7.39	7.36	7.27	**7.25**

unstable, and the performance was decreased when the input code tensor was set to a random one. Unfortunately, this slow convergence was not alleviated by including only the centroid positions of faces. In contrast, after including the facet normals to feature vectors, the convergence of the training was significantly speeded up. Furthermore, when the feature vectors include all the centroid positions, normal vectors, and areas of faces, the AADs got smaller further. Therefore, we decided to use the 7D feature vector (i.e., (p-n-a) in Tab. A.2) for NormNet.

B Hyperparameters

As we mentioned in the main text, we used different hyperparameters (i.e., the sets of weights in Eq. (10)) for CAD and non-CAD models. We compared the denoising performances of different combinations of weights in Eq. (10) to determine the hyperparameters for the different types of models. The default values for both CAD and non-CAD models are determined based on the result of hyperparameter tuning by Ray Tune [18], as we mentioned in the main text.

In this experiment, we varied the value of each weight discretely in a domain $[0.0, 5.0]$ at intervals of 0.5, while the other weights were set to the default values. Figure B.1 shows the denoising performance in AADs for different combinations of weights. Although the optimal weight values varied for different models, the performances did not differ significantly around the default values (highlighted with a black frame in Fig. B.1). Consequently, the default parameters we set are robust to different models in each category and obtain a sufficiently reasonable performance for models in both categories.

C Input Meshes and Time Performances

Figure C.1 and Tab. C.1 show all the meshes that we used in the experiments and details on them, respectively. Although our method spends more than 20

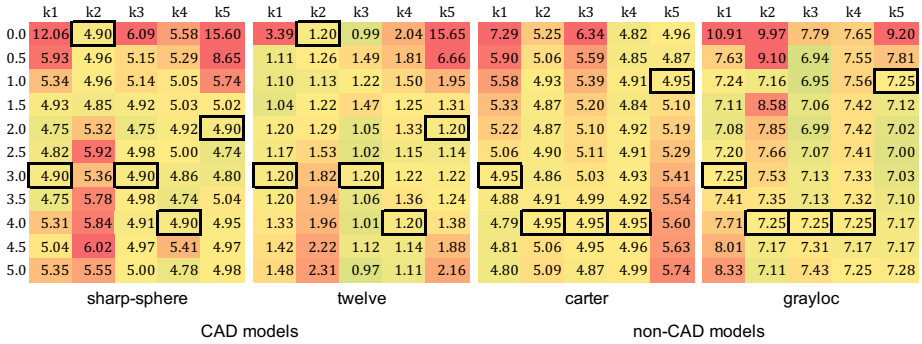

Fig. B.1. Comparison of AADs obtained by different sets of hyperparameters. For models in each of CAD and non-CAD categories, the default parameter set is highlighted by black frames.

minutes for meshes with 100 000 triangles, as shown in Tab. C.1, it does not require hours of training the neural network using a huge number of shape models. Furthermore, our method can take the meshes with such a large number of triangles directly, although many previous studies (e.g., DNF-Net [15] and GCN-Denoiser [24]) need to separate the input meshes into small patches before processing them. Hence, our computation time is sufficiently reasonable for practical use.

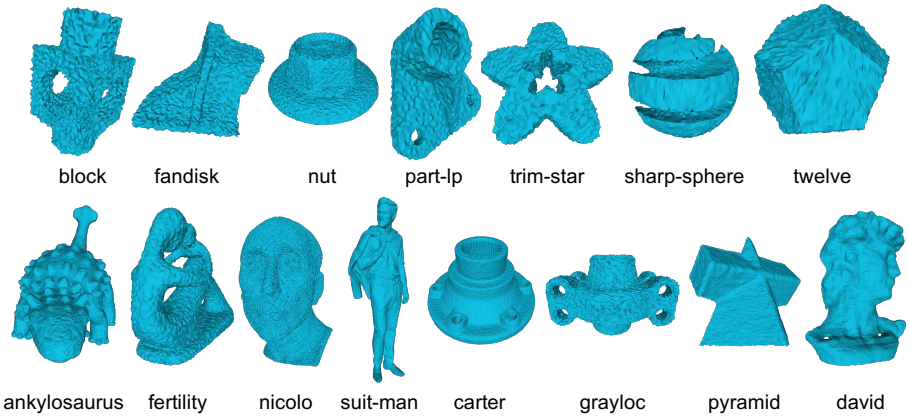

Fig. C.1. Meshes we used in the experiments. As summarized in Tab. C.1, the CAD models are shown in the top, and the non-CAD and real scan models are shown in the bottom.

Table C.1. Details on each input mesh, such as noise level, number of triangles and faces, and computation time spent by the proposed method.

Model	Type	Noise level	Vertices	Faces	Iteration	Time
block	CAD	0.4	8771	17 550	1000	4 min 42 sec
fandisk	CAD	0.3	6475	12 946	1000	3 min 42 sec
nut	CAD	0.2	7476	14 784	1000	4 min 6 sec
part-lp	CAD	0.2	4261	8530	1000	2 min 46 sec
trim-star	CAD	0.3	5192	10 384	1000	3 min 9 sec
sharp-sphere	CAD	0.2	10 443	20 882	1000	5 min 23 sec
twelve	CAD	0.2	4610	9216	1000	2 min 51 sec
ankylosaurus	non-CAD	0.1	14 762	29 520	1000	7 min 35 sec
fertility	non-CAD	0.2	13 971	27 954	1000	7 min 14 sec
nicolo	non-CAD	0.3	50 419	99 994	1000	22 min 18 sec
suit-man	non-CAD	0.1	49 996	100 000	1000	23 min 24 sec
carter	non-CAD	0.1	49 988	100 000	1000	22 min 16 sec
grayloc	non-CAD	0.3	34 274	68 580	1000	16 min 39 sec
pyramid	real scan	—	35 261	59 511	50	0 min 50 sec
david	real scan	—	51 789	101 937	50	1 min 12 sec

D Additional Results

D.1 Visual comparison with AHD

In the main text, we visually compared only the angular differences due to limitations of space. Here, Fig. D.1 shows the visual comparison based on distances between meshes. In this figure, each vertex on a mesh is colorized based on the distance to the ground-truth mesh, and the AHD to the ground-truth mesh is noted by ϵ below each mesh. Although the performances for non-CAD models of our method are approximately the same as previous methods, those for CAD models differ. Namely, our method performs better than the others, even though our method does not use a large number of training data. As shown in this figure, the distances are different, particularly at non-corner regions, which suggests the necessity of filtering vertex positions along with facet normals by the neural network, although some previous studies, such as DNF-Net [15], only filter facet normals.

D.2 Effect of iterative vertex updating

As mentioned in the main text, the consistency loss E_{con} allows PosNet and NormNet to interact with each other to be trained jointly. Thus, the iterative vertex updating performed in many previous studies is unnecessary. On the other hand, we can perform the vertex updating in postprocessing. To check the necessity of E_{con} rather than the post vertex updating, we compared their performances. As shown in Table D.1, the post vertex updating does not necessarily improve the accuracy of denoising. This result suggests that E_{con} and our end-to-end joint training of PosNet and NormNet is effective.

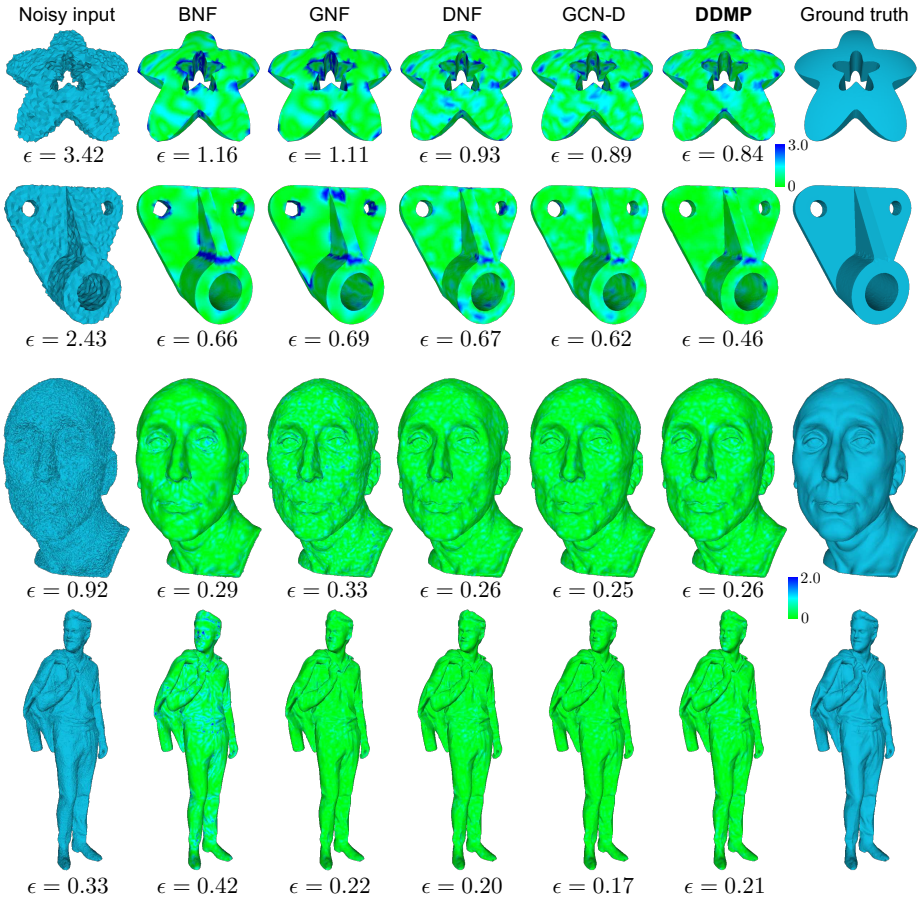

Fig. D.1. Each vertex of meshes are colorized based on the distance to the ground-truth mesh. We denote the AHD in the unit of 10^{-3} below each mesh.

	sharp	twelve	carter	grayloc
DDMP	4.90°	1.20°	4.95°	7.25°
DDMP with IVU	4.41°	0.49°	5.28°	7.40°

Table D.1. Change in AADs by adding iterative vertex updating (IVU) to our DDMP as a postprocess.