

Learning assisted Interactive Modelling with Rough Freehand 3D Sketch Strokes

Supplementary Material

Sukanya Bhattacharjee
IIT Bombay
India

sukanyabhat@cse.iitb.ac.in

Parag Chaudhuri
IIT Bombay
India

paragc@cse.iitb.ac.in

1. Method-Additional Details

1.1. Pre-processing

For the maximal plane projection, by coverage of points we mean the unique number projection of the point set that we get on a plane. We call its output as pseudo-height field when we essentially create a height field only due to taking projection on the axial planes which approximated the height field. Also, we pass the x, y values from the grid along with the value associated with it. So, the pseudo-height field is a 3D point set.

1.2. The sketchTransformer Network

We choose a transformer based architecture as we observe it to be better suited to handle unstructured and irregular data that we deal with as compared to the CNN based architectures.

Both the stage 1 and 2 decoders are linear decoders of different sizes. We found in our experiments that the linear decoders are performing better for these specific tasks as compared to convolutional decoders. In particular, the convolutional decoders output have noise in the boundary regions. The PCT based encoder outputs the positional, average and maximum feature sets corresponding to the input. The positional feature set encodes the geometry of the input. Essentially, the input points are grouped and sampled progressively in the encoder. The average feature set encodes the average geometry of each group. The maximum feature set encodes the maximum geometry of each group. For stage 1 decoder, we want to extract a denser set of points from the pseudo-height field. This requires removing unwanted points and upsampling the point set at the same time, so we use all the three feature sets. For stage 2 decoder, we want to predict the control grid from a dense set of points. This requires downsampling the point set in a way, so we use only the average feature set. We empirically found the average feature set more useful than the mean feature set for this purpose.

1.3. Post-processing

For joining the adjacent surface patches, the boundary control points are adjusted in a way that they join smoothly. In particular, we align the corresponding boundary control points and their adjacent control points from both the surfaces before stitching the boundary control points. Let this set of control points be C_B . To align the points in C_B , we first fit a cubic Bézier curve to it. Then, we further move the boundary control points such that it become the mid point of the corresponding adjacent control points from both the surfaces. Finally, we explicitly stitch the boundary control points together.

2. Training *sketchTransformer*- Additional Details

2.1. Dataset

In order to train sketchTransformer, we need a dataset with pairs of sketch strokes and surfaces. We extract BSpline surface patches and their corresponding control grids from the ABC dataset [7]. We sample random curves from these surfaces and extract points from these curves. To augment this, we extend some of these curves and add random noise to the extracted points. We re-parameterize the control grids to an empirically fixed size of 16×16 . Then we use Geomdl NURBS [2] library to recreate the BSpline surfaces from the given control points of the extracted patches. We call this dataset the *ABCSurface* dataset. From this dataset, we use 45K patches for training, 5K patches for validation and 10K patches for testing. Examples from the dataset can be seen in Figure 1.

2.2. Network details

The two stages of sketchTransformer are trained separately with an Adam optimizer with a learning rate of 0.001 for both. The Stage 2 network is trained with the output from the Stage 1 network.

Data Pre-processing: We fix the number of points to be

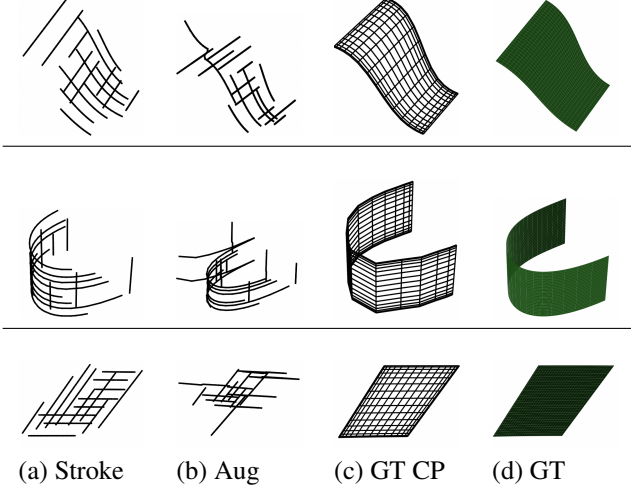


Figure 1. Example of strokes, strokes with jitter (used for augmentations), ground truth control grids and ground truth surfaces from the ABCSurface dataset.

sampled from each stroke as 20. The points and ground truth surfaces are normalized to the range $[0, 1]$. During training, we choose a random number of strokes from the given set of strokes corresponding to each strokes-surface pair.

3. Additional Results

Here we present additional results for our framework.

3.1. Comparison with Baselines

We compare our method with the baselines as given in the main paper for fitting a single surface patch to a set of sketch strokes.

3.1.1 Single Surface Patch Fitting Evaluation

We evaluate our method for fitting a single patch to a given set of 3D strokes. Figure 2 shows additional results for single surface patch fitting. The first column in the image shows the input sketch strokes, where are the rest of the columns show corresponding single surface patches fit using various methods. These methods are Least Squares Fit (LSQ) [2], ParseNet (PN) [10], StPNet (SN) [1], Variational Implicit Point Set Surface (VIPSS) [6], Neural Kernel Surface Reconstruction (NKSR) [5]. sketchTransformer (SK) is the method presented in this paper and GT is the ground truth. Figure 2 shows additional results for single surface patch fitting with our method and the baselines.

Next we test our method on a subset of the dataset with boundary strokes from the *boundABCSpline* dataset [1]. This dataset contains the boundary sketch strokes along with inner strokes in the input sketches. This is an im-

portant set-up to evaluate the effect of the availability of boundary strokes on the created surface. Table 1 and Figure 3 shows that our sketchTransformer network beats all the networks for the single patch fitting task on the bound-ABCSpline dataset as well. Figure 4 shows how the shape of the surface evolves as more sketch strokes are input.

3.1.2 Component Analysis

We also evaluate different components of the our method separately by comparing it with relevant baselines.

Stage 1: We first compare the output of this part of our framework, SK, with other baselines. We compare with the Radial Basis function and Least Square [3, 8] based approximations (RLSQ), PU Transformer [9] (PU) which is designed for upsampling of point clouds, and StPNet [1] (SN) designed for fitting surfaces to sketch strokes. Table 2 and Figure 5 presents the results for this comparison.

Stage 2: We compare this part of our framework with other baselines that perform such surface fits. This include the Radial Basis function and Least Square [3, 8] based approximations (RLSQ) and Parsenet [10] (PN). Here, we use the point set output by Stage 1 network as input as opposed to using the strokes directly. Table 3 and Figure 6 shows these results.

3.1.3 Multiple Surface Patch Fitting Evaluation

We evaluate our framework for multiple surface fitting on the standard patch based models from the classic Utah tea set dataset [11]. Quantitative and qualitative results for these can be seen in the main paper in Section 4.1.

Here, we compare average running times for various methods to fit and recover the entire model from the sampled sketch strokes. We run all the experiments on a machine with Intel Core i7 @4.20GHz CPU with 16 GB RAM, and a NVIDIA GeForce GTX 1060 6GB GPU. These results are presented in Table 4. Note that these are times for the entire models, over multiple patches and even though our method is slower than NKSR, we produce much better results as can be seen in results in the main paper and in Figure 7. Also, our method is still fast enough to be suitable for interactive modelling.

Figure 7 shows spatial distribution of the error on point samples taken from patches fit using various methods for the same evaluation for the comparison given in Section 4.1 Figure 6 in the main paper.

3.2. Modelling from real 3D sketches

3.2.1 Results on Existing Dataset

We show additional examples of surface models created from 3D curated dataset [4] in Figure 8.

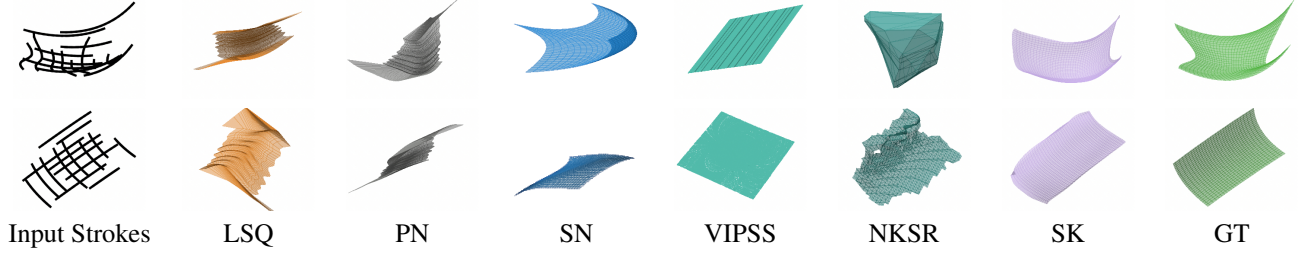


Figure 2. Additional results for comparing with baselines for creating single surface patches from input strokes, on ABCSurface dataset. **LSQ**: [8], **PN**: [10], **SN**: [1], **VIPSS**: [6], **NKSR**: [5], **SK**: Our method.

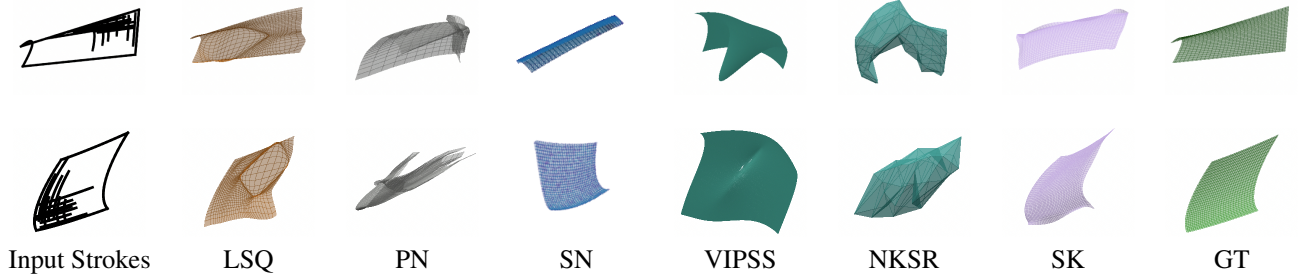


Figure 3. Comparing with baselines for creating single surface patches from input strokes, on boundABCSpline dataset. **LSQ**: [8], **PN**: [10], **SN**: [1], **VIPSS**: [6], **NKSR**: [5], **SK**: Our method.

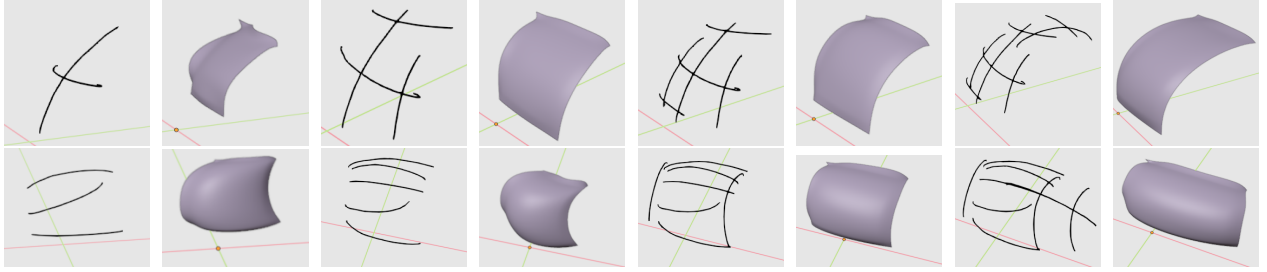


Figure 4. Evolution of shape of the surface fit by our method with increasing number of strokes.

Chamfer						Hausdorff					
LSQ	PN	SN	VIPSS	NKSR	SK	LSQ	PN	SN	VIPSS	NKSR	SK
0.0897	0.0437	0.299	0.1438	0.1216	0.0103	0.3254	0.5067	1.0679	0.4859	0.4226	0.305

Table 1. Comparing with baselines for creating single surface patches from input strokes, on boundABCSpline dataset. **LSQ**: [8], **PN**: [10], **SN**: [1], **VIPSS**: [6], **NKSR**: [5].

	RLSQ	PU	SN	SK
Chamfer	1.4e+13	0.0214	0.3786	0.0119
Hausdorff	0.4929	0.2778	1.1009	0.2278

Table 2. Comparison of our Stage 1 network with the respective baseline methods on ABCSurface dataset. The percentage improvement of our method over the best performing baseline are: Chamfer: **44.39**, Hausdorff: **17.99**. **RLSQ**: [3, 8], **PU**: [9], **SN**: [1], **SK**: Our proposed method

	LSQ	PN	VIPSS	NKSR	SK
Chamfer	0.4046	0.0478	0.3705	0.43	0.0235
Hausdorff	0.3132	0.4557	0.3095	0.3729	0.2989

Table 3. Comparison of our Stage 2 networks with the respective baseline methods on ABCSurface dataset. The percentage improvement of our method over the best performing baseline are: Chamfer: **50.84**, Hausdorff: **3.42**. **LSQ**: [8] **PN**: [10], **VIPSS**: [6], **NKSR**: [5], **SK**: Our proposed method

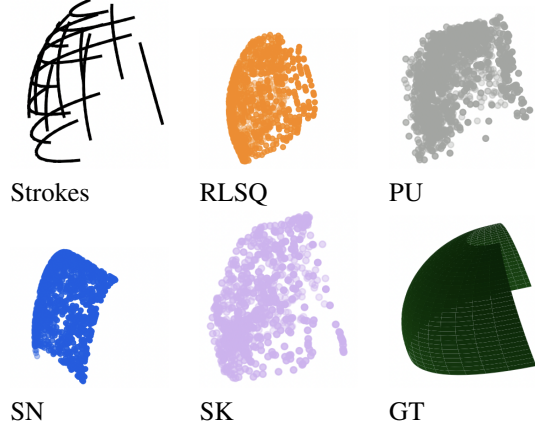


Figure 5. Qualitative comparison of our Stage 1 shape preserving transformation of sketches to a uniform and regular set of points with the baseline methods on ABCSurface dataset. **RLSQ**: [3, 8], **PU**: [9], **SN**: [1], **SK**: Our proposed method.

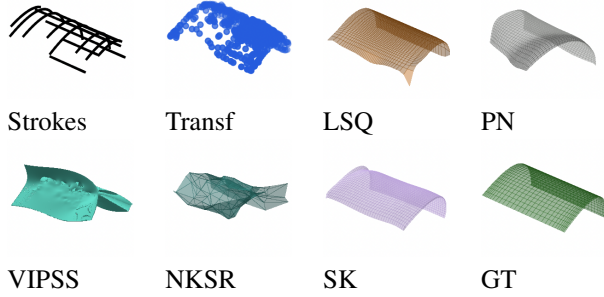


Figure 6. Qualitative comparison of our Stage 2 surface fitting from our shape preserving transformation with the baseline methods on ABCSurface dataset. **Transf**: Stage 1 network output for given input, **LSQ**: [8], **PN**: [10], **VIPSS**: [6], **NKSR**: [5], **SK**: Our proposed method.

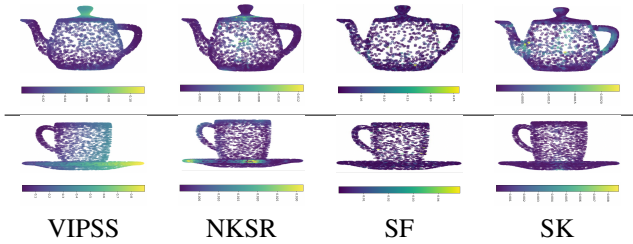


Figure 7. Error maps corresponding to the reconstructions by the baselines and our method. The error bar below each error map shows the range of error. **VIPSS**: [6], **NKSR**: [5], **SF**: [12], **SK**: Our method.

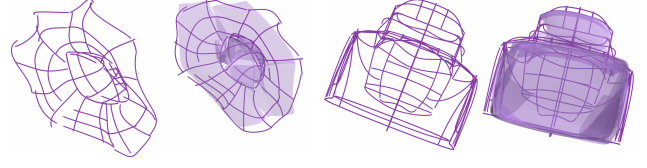


Figure 8. Additional examples of surface models created with our method for sketches from 3D curated dataset [4]. We show input sketch strokes and the predicted surface models overlaid with the sketch strokes.

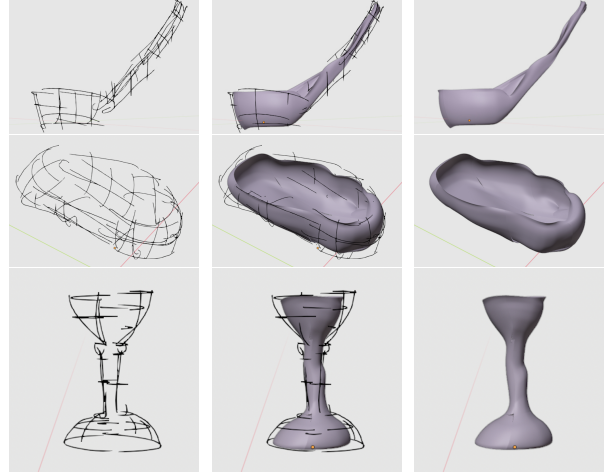


Figure 9. Additional results of creating different 3D models from real-time 3D sketches with our Blender plugin.

	Teapot	Teacup	Teaspoon
VIPSS	1878.0	1878.0	1878.0
NKSR	0.97	0.90	0.73
SF	240	230	200
SK	26.2	21.9	13.95

Table 4. Average running time (in seconds) for surface modelling on the Utah tea set dataset [11] (Teapot, Teacup and Teaspoon models). **VIPSS**: [6], **NKSR**: [5], **SF**: [12], **SK**: Our method.

3.2.2 Results of The Blender Plugin Frontend

We show additional examples of surface models created from real-time 3D sketches with our Blender plugin frontend in Figure 9.

3.3. User Study

Figures 10 and 11 show the pairs of sketch and surface models that we show to the participants in the user study.

4. Limitation

Our method is not able to successfully deal with cases where the sketch strokes have sharp curves (see Figure 12).

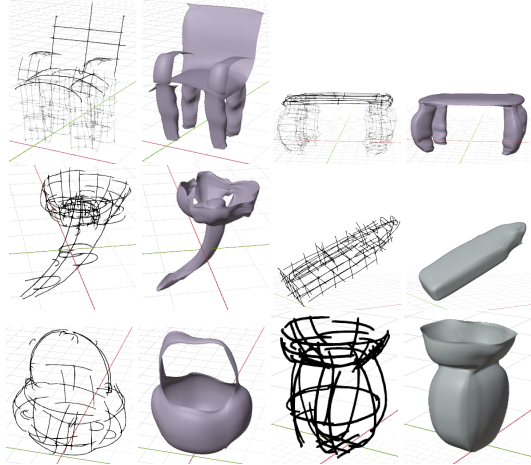


Figure 10. Pairs of sketch and models shown to the participants in Blender during the user trials.

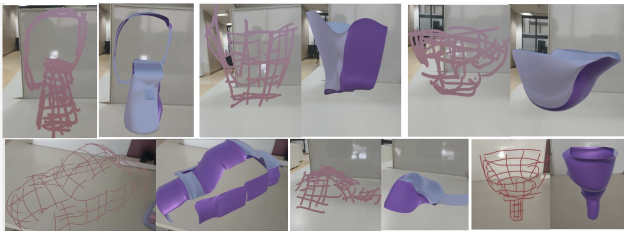


Figure 11. Pairs of sketch and models shown to the participants in Mobile AR during the user trials.



Figure 12. Our method does not produce correct patch fit for strokes with sharp creases. The middle image shows the incorrect result produced from the sketch strokes in the first image, and the final image shows the corrected result produced by splitting the strokes at the crease.

This is due to the reason that multiple points are projected to same point on the maximal projection plane.

This can be solved by splitting the strokes at the sharp curvature and fit two different patches to them.

References

- [1] Sukanya Bhattacharjee and Parag Chaudhuri. Deep interactive surface creation from 3d sketch strokes. In *Proceedings of IJCAI*, pages 4908–4914. International Joint Conferences on Artificial Intelligence Organization, 2022. AI and Arts Track. 2, 3, 4
- [2] Onur Rauf Bingol and Adarsh Krishnamurthy. NURBS-Python: An open-source object-oriented NURBS modeling framework in Python. *SoftwareX*, 9:85–94, 2019. 1, 2
- [3] Martin D Buhmann. Radial basis functions. *Acta numerica*, 9:1–38, 2000. 2, 3, 4
- [4] Emilie Yu. 3d sketches curated dataset, 2023. <https://gitlab.inria.fr/D3/3d-sketches-curated-dataset> Last accessed on 24-02-2024. 2, 4
- [5] Jiahui Huang, Zan Gojcic, Matan Atzmon, Or Litany, Sanja Fidler, and Francis Williams. Neural kernel surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4369–4379, 2023. 2, 3, 4
- [6] Zhiyang Huang, Nathan Carr, and Tao Ju. Variational implicit point set surfaces. *ACM Transactions on Graphics (TOG)*, 38(4):1–13, 2019. 2, 3, 4
- [7] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9601–9611, 2019. 1
- [8] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 1996. 2, 3, 4
- [9] Shi Qiu, Saeed Anwar, and Nick Barnes. Pu-transformer: Point cloud upsampling transformer. In *Proceedings of ACCV*, pages 2475–2493, 2022. 2, 3, 4
- [10] Gopal Sharma, Difan Liu, Subhransu Maji, Evangelos Kalogerakis, Siddhartha Chaudhuri, and Radomír Měch. Parsenet: A parametric surface fitting network for 3d point clouds. In *European Conference on Computer Vision*, pages 261–276. Springer, 2020. 2, 3, 4
- [11] Ann Torrence. Martin Newell’s original teapot. In *ACM SIGGRAPH 2006 Teapot*, pages 29–es. 2006. 2, 4
- [12] Emilie Yu, Rahul Arora, J Andreas Baerentzen, Karan Singh, and Adrien Bousseau. Piecewise-smooth surface fitting onto unstructured 3d sketches. *ACM Transactions on Graphics (TOG)*, 41(4):1–16, 2022. 4