Pix2Surf: Learning Parametric 3D Surface Models of Objects from Images — Supplementary Material

Jiahui Lei[1] Srinath Sridhar[2] Paul Guerrero[3] Minhyuk Sung[3]
Niloy Mitra[4,3] Leonidas J. Guibas[2]

[1]Zhejiang University [2]Stanford University [3]Adobe Research
[4]University College London

https://geometry.stanford.edu/projects/pix2surf

S.1 Overview

In this supplementary, we provide additional details about our training (Sec. S.2) and inference setups (Sec. S.3), and details of our evaluation metrics (Sec. S.4). We provide an extended qualitative comparison of our method to the Image2Surf baseline (Sec. S.5), ablations (Sec. S.6) and for visible surface generation on real-world data (Sec. S.7). We show additional qualitative results for hidden surface generation (Sec. S.9) and also provide more visual results for Pix2Surf (Sec. S.8) and more qualitative comparison to Pixel2Mesh++[3] and AtlasNet[1] (Sec. S.10).

S.2 Training Details

For the **Single-View** case, we train our network in two phases. In the first phase, we train the NOCS-UV branch with a learning rate of 1e−4, using the NOCS Map and the object mask as supervision. In the second phase, we add the remaining SP branch and train end-to-end until convergence, with a learning rate of 1e−4 for cars and 3e−5 for planes and chairs, and using the losses described in Sec. 4.1 in the paper.

For the **Multi-View** case, we have found that pre-training with the single-view architecture, before switching to the full multi-view architecture results in better initialization. For this purpose, we start by passing the feature z_m, directly to the SP branch without max-pooling multiple views. After pre-training, we switch to the multi-view architecture as described in Sec. 4.2 in the paper, by max-pooling the z_m features of all views, and concatenating both this max-pooled multi-view feature, and the single-view feature z_m for the current view as input to the MLP. To better fuse multi-view information for learned chart prediction, the feature map in the middle of CNN encoder and decoder also follows above fusion operation. We randomly pick 5 views as input during multi-view training. For our multi-view consistency loss, we need to identify corresponding pixels in different views. We sample pixels in each view as in the single-view case and find

corresponding pixels based on their distance in NOCS coordinates. Two pixels are in correspondence if their NOCS distance is less than 1e−3.

We separately train on each object category of our dataset.

S.3 Inference Details

One significant advantage of our explicit **continuous** parametric surface prediction is that we can sample the results at any resolution (e.g. points or vertices). We generate our final predictions at a regular grid of samples in the unwrapped uv chart, obtaining a 3D location for each sample (obtained from the SP-Branch). Since we have exact correspondence to pixels of the input image, each sample also has a color value (or interpolated color value in super-resolution). Samples corresponding to background pixels are masked out. To create a mesh, we can connect neighboring foreground samples with edges. All visual results of our method in the paper are generated using this approach. We provide more details.

Identifying foreground regions in the unwrapped chart. Unlike AtlasNet, the shape and topology of the unwrapped surface in our chart is learned by the NOCS-UV branch, which gives the reconstructed surface more flexibility to represent arbitrary shapes and topologies. To identify foreground regions in the uv space of the unwrapped chart, we map the the learned image-space foreground mask to uv space. Directly unwrap the mask by learned-uv map (two channel output from NOCS-UV branch) results in pixel cloud with holes in uv space. To solve this issue, we up-sample the image-space mask and learned-uv map from its original resolution of 240×320 by a factor of 4 using linear interpolation before mapping mask to uv space. To avoid interpolating across C^0 discontinuities of the surface, we only interpolate neighboring pixels that are mapped to similar uv locations (i.e., the gradient of their uv coordinates is below a threshold). We then map the up-sampled mask to uv space (resolution of 128×128) by the up-sampled learned-uv map. Finally we up-sample the mask in uv space to the desired resolution (in paper we use 512×512).

In uv space, we additionally post-process the unwrapped foreground mask by closing small holes using morphological operations. Finally, we remove outliers using the predicted 3D locations (quarried from SP-Branch) of each mask sample. A sample of the foreground mask is classified as outlier if the distance in 3D space to its nearest neighbor is larger than a threshold t. In practice, we use $t = 0.03$ for chairs and $t = 0.02$ for cars and airplanes. Similar outlier removing operation is also applied to image-space mask before identifying foreground regions.

Texturing the unwrapped chart. Similar to the mask, directly unwrapping the image-space color values to the uv space results in a sparse set of irregular color samples in uv space. We can interpolate these samples to obtain the color value at any point in uv space by interpolating the k nearest neighbors (we use $k = 4$ for our results).

S.4 Evaluation Metrics

We now define the evaluation metrics used in the paper.

A common surface representation: Before evaluating our metrics, we convert the results of all methods to a common format to avoid biasing our results due to different surface representations. We convert all output representations to the NOCS-Map format defined in X-NOCS [2] using the ground truth camera model. The NOCS map $\mathcal{P}$ samples the reconstructed surface from a single viewpoint, giving a point cloud where each sample has a 2D pixel coordinate p and a 3D location x. The 3D location is defined in a canonical coordinate frame that is shared across views and across instances of the same shape category. For multi-view reconstructions, we create one NOCS-Map for each viewpoint, compute the metrics on each NOCS-Map, and average the results over all views. As AtlasNet [1] ground truth is not in the same ShapeNet version as ShapeNet-Plain [2], we first scale the AtlasNet results to have the same bounding box diagonal as the ground truth 2-intersection X-NOCS maps point cloud, and then align the lower left corner of the bounding box.

The **Reconstruction Error** is measured as the 2-Way-Chamfer-Distance between the ground truth NOCS-Map $\mathcal{P}_1$ and predicted NOCS-Map $\mathcal{P}_2$:

$$E_{\text{rec}} = \frac{1}{|\mathcal{P}_1|} \sum_{x_i \in \mathcal{P}_1} \min_{y_j \in \mathcal{P}_2} \|x_i - y_j\|_2^2 + \frac{1}{|\mathcal{P}_2|} \sum_{y_j \in \mathcal{P}_2} \min_{x_i \in \mathcal{P}_1} \|x_i - y_j\|_2^2.$$

The reconstruction error for hidden surfaces in Table 2 of paper is computed in the same way, but using NOCS-Maps of the hidden surfaces.

The **Correspondence Error** is measured as the squared distance between the predicted 3D location x_i and the ground truth location y_i of the same pixel:

$$E_{\text{corr}} = \frac{1}{|\mathcal{M}|} \sum_{p_i \in \mathcal{M}} \|x_i - y_i\|_2^2.$$

We only evaluate pixels $p_i \in \mathcal{M}$ that are both in the predicted and ground truth foreground masks.

Consistency Error is based on the squared distance between the predicted 3D locations of corresponding pixels in different views. For each pair of views a and b, we identify corresponding pairs of pixels (p_i^a, p_j^b) as pairs having a similar ground truth 3D location in NOCS: $\|y_i^a - y_j^b\|_2 < \epsilon$. In practice, we set $\epsilon = 0.001$. We then average the squared distance between the predicted 3D locations x_i^a and x_j^b of all corresponding pixel pairs $\mathcal{P}_{\text{corr}}^2$:

$$E_{\text{cons}} = \frac{1}{|\mathcal{P}_{\text{corr}}^2|} \sum_{(p_i^a, p_j^b) \in \mathcal{P}_{\text{corr}}^2} \|x_i^a - x_j^b\|_2^2.$$

With the **Discontinuity Score**, we take a statistical approach to measure the correctness of the surface connectivity. While the continuity of implicit or parametric surface is a property induced by representation and method design,

we need to make sure the continuity is correct, i.e. no over-smooth results. We compute statistics of the C^0 discontinuities in the predicted surface, and measure the similarity to the same statistics computed on the ground truth surface. The statistics are based on the 3D distance $\|x_i - x_j\|_2$ of neighboring foreground pixels p_i and p_j. Pixels with a large difference are likely to lie on the border of a C^0 discontinuity of the predicted surface. We compute a histogram h of this 3D distance over all neighboring pixels:

$$h_i = |\{(p_i, p_j) \in \mathcal{P}^2_{\text{neighbors}} \mid t_i \leq \|x_i - x_j\|_2 < t_{i+1}\}|,$$

where t_i are the boundaries of the histogram bins and $\mathcal{P}^2_{\text{neighbors}}$ is the set of all neighboring pixel pairs. We use a 4-connected neighborhood and choose 20 bins with bin edges spaced uniformly in $[0.05, \sqrt{3}]$. We measure the similarity of two histograms as the correlation of their normalized bins:

$$S_{\text{cont}} = \frac{1}{\sum_k h_k \sum_j h^{\text{gt}}_j} \sum_i h_i h^{\text{gt}}_i$$

Note that unlike the other errors we use as evaluation metrics, this is defined as a score, where higher values imply more accurate discontinuity of the reconstructed surface.

S.5 Qualitative Comparison to Image2Surf

We show more qualitative comparisons between our baseline Image2Surf and Pix2Surf in paper Sec. 5.1. Image2Surf has a fatal problem to make "cut" around the occlusion boundary (i.e., wrong C^0 discontinuities), which is reflected both in the red rectangle in Fig. S1 and discontinuity score in Table 1 in paper.

S.6 Ablations

We provide an analysis and justification of several key design choices. First, we analyze the importance of using a learned UV chart instead of a fixed chart like

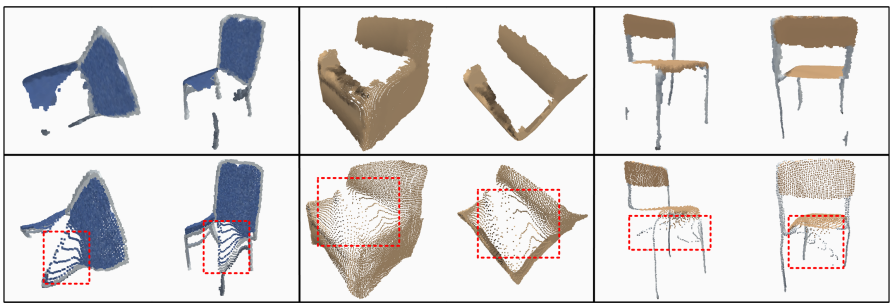


Fig. S1. Qualitative Comparison to Image2Surf. The first row are the results of Pix2Surf and the second row are for Image2Surf. Each instance is viewed from 2 different viewpoints. Image2Surf wrongly connects disjoint parts and results in strong distortions, which are solved by Pix2Surf's learned chart.

Image2Surf. As seen in Sec. 5.1 and Table 1 in the paper, Pix2Surf outperforms Image2Surf on all categories for reconstruction error. Second, we analyze the utility of multi-view feature pooling and consistency loss. As seen in Table 1 (rows 4–7), these two features significantly improve performance. We also justify the use of intermediate NOCS map regression by the NOCS-UV branch, and the need for the UV amplifier (Table S1). We do so by examining networks without these two components. For the NOCS map ablation, we train the network from scratch without pretraining the NOCS-UV branch, and for the UV amplifier ablation, we directly input the learned UV coordinates to the SP branch and increase the dimension of a latent image code to 256. When conducting the experiments on the chair category, the results (Table S1) show that these components help learn better reconstructions.

Table S1. We experimentally verify the usefulness of NOCS map regression and the UV amplifier. NOCS map regression provides intermediate supervision while the UV amplifier balances information. Here we report average reconstruction error computed on the *visible* part (equal training epochs for all methods).

	No UV Amp.	No NOCS	Pix2Surf
Chair	10.37	3.64	**2.61**

S.7 Qualitative Results on Real-World Data

We show more results for generalization to real world data mentioned in paper. In Fig. S2, we show single- and multi-view results for Pix2Surf that is trained on ShapeNet COCO and inference on real world car. Note that the texture in each view separately is better than the multi-view aggregation. This is caused by the different light condition from different viewpoints. As our main concern in this paper is not to fuse the texture from multiple views, we leave the improvement of the texture to future works.

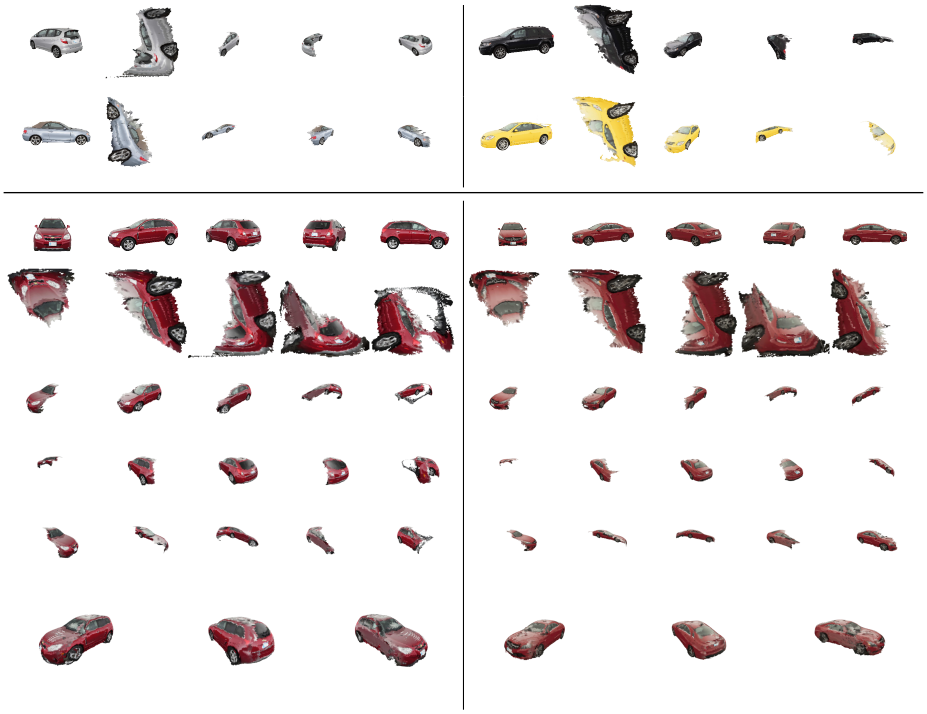


Fig. S2. Real world image generalization. The top part is single-view visualization: input image, unwrapped chart with texture and 3 viewpoints of the reconstruction for each instance. The bottom part is multi-view aggregation visualization. For every instance, each row is: input images, unwrapped charts with texture, 3 viewpoints for each view's result separately and finally multi view aggregation.

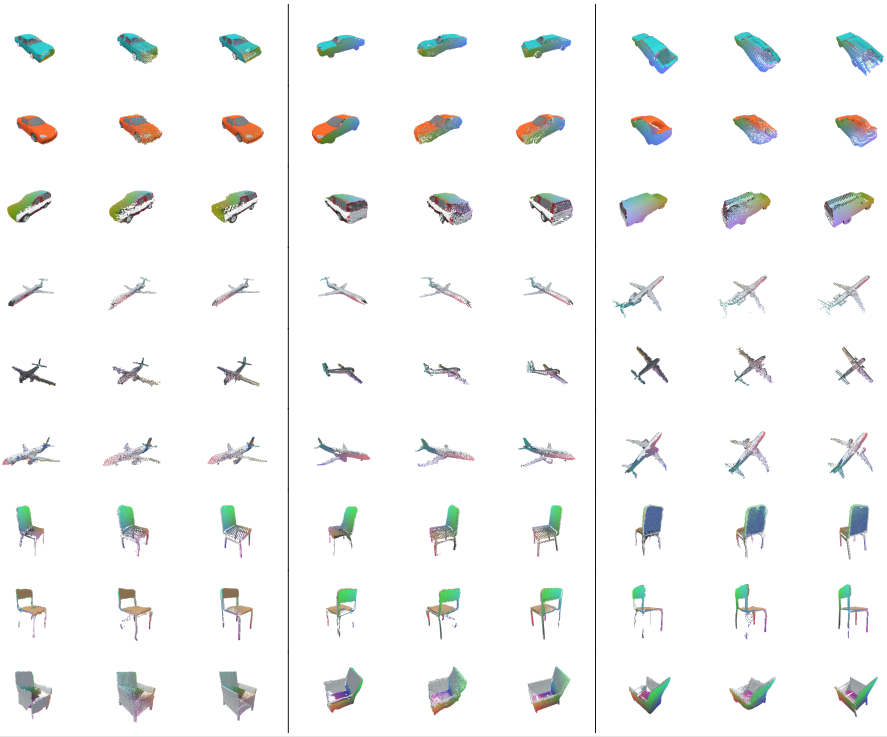
S.8 More Results

Figure S3 shows more results of Pix2Surf including the learned UV map (as shown in Figure 4) and reconstruction outputs of both single-view and multi-view architectures. See the caption for the details.

Fig. S3. Single-view and multi-view Pix2Surf reconstruction results. The results for each object are presented in three rows. The first row shows five input views. Note that we do not have camera parameters for any of these views. The second row shows the per-view UV space that is generated by the multi-view variant of Pix2Surf. The UV space is not directly constrained by any loss; the flattening of the objects that we can observe and the large degree of consistency between different views is an emergent property of our network. In the third row, we show, from left to right, (a) the reconstructed 3D surface obtained by merging Pix2Surf single-view reconstructions (SV), (b) the Pix2Surf multi-view reconstruction (MV), and (c) the ground truth reconstruction (GT). The last three columns show the same results from a different viewpoint. Note the reduction in the number of gaps and surface discontinuities when comparing the multi-view to the single-view results.

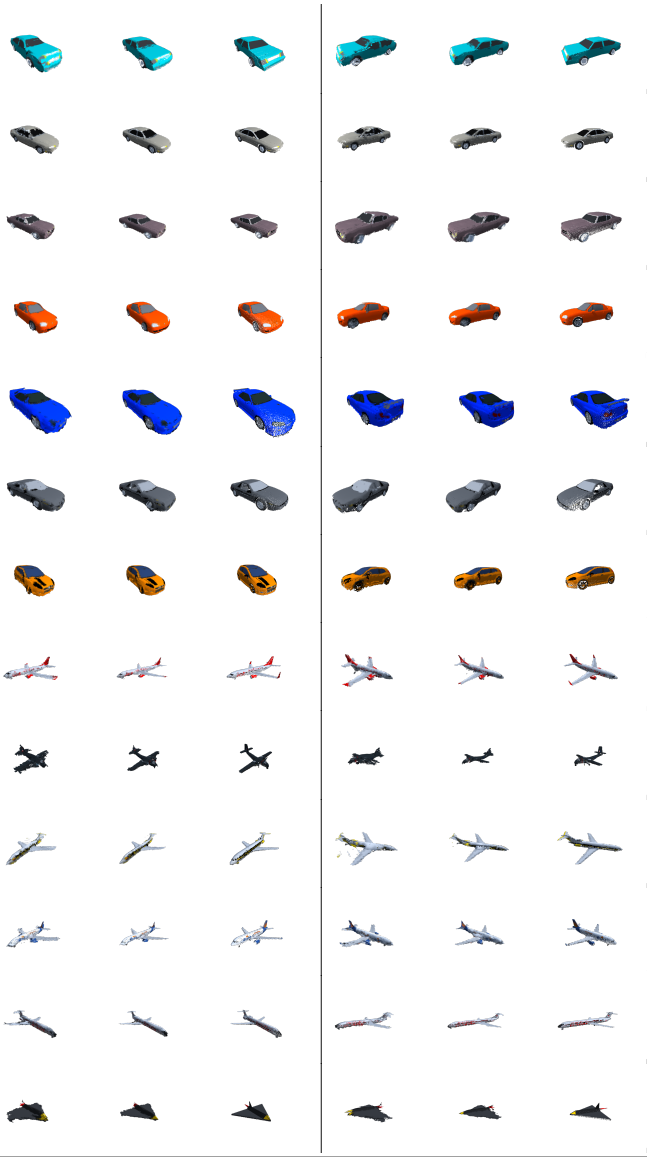
S.9 Qualitative Results for Hidden Surface Generation

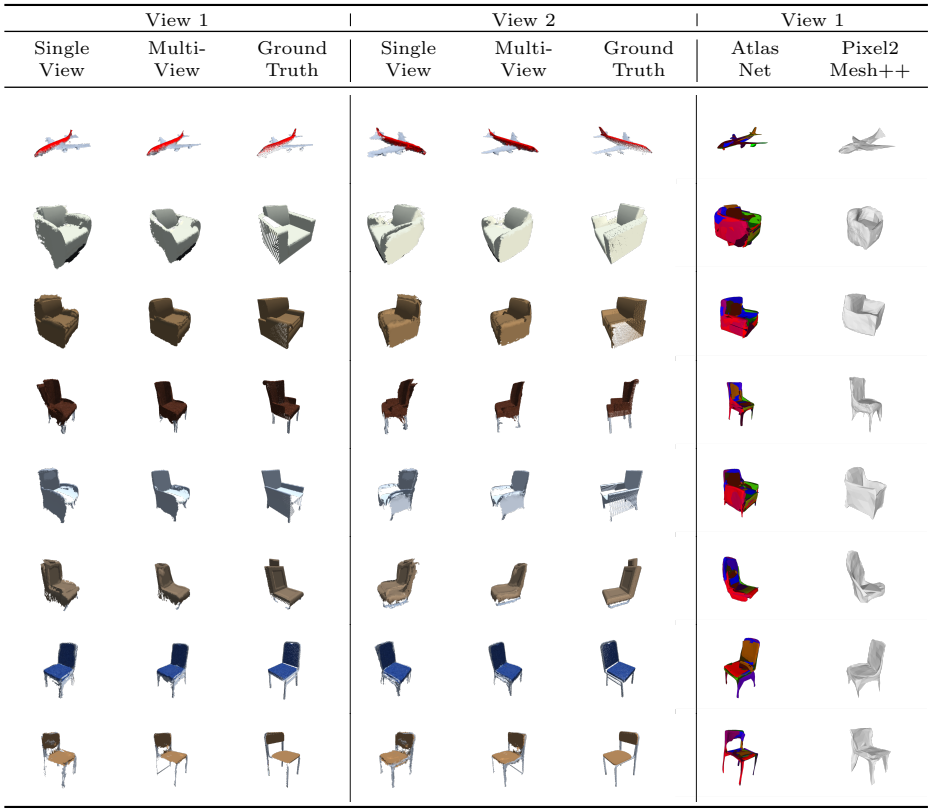
The following table provides more visual results of Pix2Surf Two-Intersection version (Sec 5.2 in paper), and comparison with X-NOCS [2]. Pix2Surf can easily be extended to capture the invisible surface and is more accurate and smooth than X-NOCS.

View 1			View 2			View 3		
Pix2Surf (sv)	X-NOCS (sv)	Ground Truth	Pix2Surf (sv)	X-NOCS (sv)	Ground Truth	Pix2Surf (sv)	X-NOCS (sv)	Ground Truth
								

S.10 Qualitative Comparisons

The following table demonstrates qualitative comparisons among our Pix2Surf (both single-view and multi-view architectures), AtlasNet[1], and Pixel2Mesh++ [3]. The colors in AtlasNet results show different output patches.

View 1			View 2			View 1	
Single View	Multi-View	Ground Truth	Single View	Multi-View	Ground Truth	Atlas Net	Pixel2 Mesh++
							


View 1
View 2
View 1
Single View
Multi-View
Ground Truth
Single View
Multi-View
Ground Truth
Atlas Net
Pixel2 Mesh++

References

1. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: Atlasnet: A papier-mâché approach to learning 3d surface generation. In: Proc. of CVPR (2018)
2. Sridhar, S., Rempe, D., Valentin, J., Bouaziz, S., Guibas, L.J.: Multiview aggregation for learning category-specific shape reconstruction. In: Proc. of NeurIPS (2019)
3. Wen, C., Zhang, Y., Li, Z., Fu, Y.: Pixel2mesh++: Multi-view 3d mesh generation via deformation. In: Proc. of ICCV (2019)