Figure 9: **Additional Results.** We show results of our method on additional diverse meshes. Our method automatically predicts a fine-grained segmentation mask that accurately reflects a plausible placement of the reference image on the mesh (*e.g.* the overalls cover the legs and go over the shoulders and the scarf is draped around the neck). These results contextualize the reference image and synthesize a plausible variant of it that conforms to the input geometry and to the local predicted segmentation mask (see for example the colorful poncho on the left). Our method also captures the texture detail in the overall jeans (see for example the green line present in the result).

A Robustness

Fig. 10 demonstrates the robustness of our method to the localization initialization. The local region can start from a given specific shape different than that of the reference image object and still converge to a region corresponding to the asset in the guidance image.

Figure 10: **Robustness.** 3D PixBrush is robust to the initial localization region. Here, the localization starts out as heart-shaped sunglasses (left), which is different than the shape of the sunglasses in the guidance image examples shown. Still, our method overcomes this discrepancy and corrects it to match the reference image.

B Limitations

In Fig. 11, we show the limitations of our method. 3D PixBrush struggles to reproduce text depicted in the guidance image. Additionally, there may be semantic coupling in the edited result, where regions of the mesh closely related semantically to the asset in the image are colored as well.

Figure 11: **Limitations.** Our method captures the style of the reference image but can struggle to exactly reproduce text (left). In cases where the guiding image carries strong semantic connotations, components that are closely semantically related to the guiding image, but not included in the image can be included in the localization (right). For example, in this image that contains makeup, lipstick is also included in our edit.

C Additional Quantitative Experiments

We show additional quantitative comparisons between our method and 3D Paintbrush using a perceptual study, additional CLIP metrics, and the LPIPS metric.

Perceptual Study. We conduct a perceptual study in which 34 users rate the effectiveness of both our method and 3D Paintbrush on 10 different mesh and reference image combinations. The users are asked to evaluate the quality of both the edit texture (the actual colors and patterns) and edit structure (the shape and location on the object). 3D PixBrush (ours) scores higher than 3D Paintbrush, producing more accurate textures and structures (see Tab. 2).

Table 2: Perceptual Study. We conduct a perceptual study where users evaluate the texture and structure of our local edits compared to 3D Paintbrush Decatur et al. (2024).

Method	Texture Avg. Score ↑	Structure Avg. Score ↑
3DPB	1.96	2.69
Ours	**4.11**	**4.17**

Text-based CLIP R-Precision. In addition to the image-based CLIP R-Precision that we report in the main paper, we run a text-based CLIP R-Precision evaluation as well (see Tab. 3). Instead of using the renders of the generated results to retrieve the reference images, we use the renders to retrieve the BLIP 2-generated captions. We find that our method still outperforms 3D Paintbrush Decatur et al. (2024) despite the fact that our method is only conditioned on the reference images and does not take the captions as input, while 3D Paintbrush is *directly* conditioned on these exact BLIP 2 captions. We hypothesize that this is due to the fact that text-driven approaches like 3D Paintbrush don't capture all sentiments contained in the text prompt equally well and tend to neglect certain components. In contrast, our image-driven approach better captures the global style of the reference image. Since the BLIP 2 captions (which are used as text prompts) are derived from the reference images, 3D PixBrush captures more components of the text prompts.

Table 3: Text-based CLIP R-Precision. We compare our local edits to 3D Paintbrush Decatur et al. (2024) (3DPB) and report a text-based CLIP R-Precision metric using BLIP 2 Li et al. (2023a) captions of the reference images. Our method outperforms 3D Paintbrush on all CLIP models, despite the fact that our method is guided by the reference images, whereas 3D Paintbrush directly uses these BLIP 2 captions as text prompts to guide their optimization.

	Text-based CLIP R-Precision ↑		
Method	CLIP B/32	CLIP B/16	CLIP L/14
3DPB	76.19	76.19	80.95
Ours	**90.48**	**80.95**	**85.71**

CLIP Similarity Metric. In addition to the R-Precision metrics, we compare our method to 3D Paintbrush Decatur et al. (2024) by directly reporting average CLIP similarity scores in Tab. 4. We take the renders of the generated results for each image and encode them into CLIP's latent space. We also encode the reference images. We then compute the cosine similarity in CLIP latent space between the generated renders and their corresponding reference images and report the average similarity over all examples. 3D PixBrush achieves higher similarity scores than 3D Paintbrush on all CLIP models.

LPIPS Metric. In Tab. 5, we evaluate our method as compared to 3D Paintbrush using the LPIPS Zhang et al. (2018) metric. This metric uses deep features of visual models to measure similarity between images. Here, lower scores correspond to higher similarity. We find that 3D PixBrush achieves lower LPIPS scores and thus higher similarity with the reference images than 3D Paintbrush using deep features from both AlexNet Krizhevsky et al. (2012) and VGG Simonyan (2014).

Table 4: CLIP Similarity Scores. We compare our local edits to 3D Paintbrush Decatur et al. (2024) (3DPB) and report the average cosine similarity in CLIP embedding space between renders of the generated results and their corresponding reference images. Our method receives higher average similarity scores than 3D Paintbrush on all CLIP models.

	CLIP Similarity Scores ↑		
Method	CLIP B/32	CLIP B/16	CLIP L/14
3DPB	0.61194	0.62860	0.57538
Ours	**0.65325**	**0.66936**	**0.61043**

Table 5: LPIPS Quantitative Evalutation. We compare our local edits to 3D Paintbrush Decatur et al. (2024) (3DPB) and report the LPIPS Zhang et al. (2018) perceptual metric between the generated results and their corresponding reference images. Our method outperforms 3D Paintbrush on both AlexNet and VGG.

Method	LPIPS AlexNet ↓	LPIPS VGG ↓
3DPB	0.63988	0.52314
Ours	**0.63575**	**0.51803**

D Implementation Details and Ablations

Supervision and Optimization Details. For our localization loss, we employ DeepFloyd IF AI (2023) and run it for 1000 optimization steps. We then continue to update the localization region as well as optimize the local texture edit according to the reference image using both DeepFloyd and IP-Adapter Plus Ye et al. (2023) for an additional 10000 steps. During these additional 10000 iterations, we still use the localization loss as well. 3D PixBrush is implemented in PyTorch Paszke et al. (2017). We run our experiments using an NVIDIA A40 GPU. For the results shown in the paper, we run our local texture optimizations for 4 hours, however, we obtain reasonable results after roughly 1 hour (see Fig. 12).

View Selection Details. In order to supervise our optimization, we render 2D images of our 3D mesh. The camera parameters for these renders are randomly sampled each iteration of the optimization. By default, camera elevation values are sampled between 0 and 60 degrees, azimuth values are sampled from the range $0 - 360$ degrees, and radius values are sampled from the range $1 - 1.5$. For a small subset of meshes that are predominantly viewed from above, we expand the elevation range to $0 - 150$.

Neural Network Architecture Additional Details. As specified in Sec. 3.1, we use MLPs consisting of a positional encoding layer followed by 6 fully connected layers for both the neural localization and neural texture networks. The positional encoding layer in the neural localization network uses a σ value of 6 while the positional encoding in the neural texture network uses a σ value of 12 to allow for the higher frequency textures we output. The middle layers of the MLPs have width 256

Ref. image 1 hour result 4 hour result

Figure 12: **Optimization Run Time.** We show the results of our approach after running the optimization for both 1 hour (middle) and a full 4 hours (right). In many cases, our approach achieves satisfactory results after only 1 hour that are comparable to the full 4 hour results.

Figure 13: **Localization Loss Ablation**. 3D PixBrush (Ours) produces accurate localizations and detailed textures, each of which matches the structures and styles respectively of the reference images. Removing the localization loss (w/o loc loss) often results in trivial localizations that segment all of the mesh and global textures that ignore the geometry.

Figure 14: **Composition of Local Texture Edits.** 3D PixBrush produces precise localizations to accompany the predicted local textures. A key strength of predicting an *explicit fine-grained segmentation mask* is that it enables an additional level of control for graphics applications. For example, it is possible to trivially composite multiple local textures on the same mesh. Another application of our predicted localization masks is shown in Fig. 15.

and the output layers produce 1 value (probability) for the localization network and 3 values (RGB) for the texture network.

Localization Loss Ablation. In Fig. 6 in the main paper, we show that removing the localization loss warm up period degrades performance. Here, we further investigate the importance of the localization loss by removing it entirely. In Fig. 13, we see that removing the localization loss entirely still results in degraded performance: either all or none of the region is localized, and textures are applied globally instead of locally.

E Additional Results

Composition of Local Texture Edits. In Fig. 14, we show composited local textures using 3D PixBrush. Since our method produces precise localizations that accompany our local textures, we can easily composite one texture on top of another. We first locally texture the person to add the dress shirt (middle left). Then we independently locally texture the person to add the sweater vest (middle right). Using the localizations for each respective local texture, we composite the sweater texture on top of the dress shirt texture and add both to the person in a coherent way (far right).

Edits Over Existing Textures. In Fig. 15, we use our method to generate local textures on a bare mesh and overlay them on an existing global texture (left) for that mesh. The precise localization masks predicted by our method enable us to trivially composite our local textures onto the mesh such

Figure 15: **Compositing Over Existing Textures.** Starting with an existing global texture (left), we can use our precise localization masks to trivially composite our local textures onto the mesh. Our predicted localization masks are precise enough that they enable effectively matting the predicted texture while leaving the original cow texture unchanged. This application is trivially enabled due to our deliberate choice in predicting a fine-grained segmentation mask.

that we only change regions corresponding to the local texture and preserve the rest of the existing global texture on the mesh (middle, right).

Same Reference Image Applied to Diverse Meshes. In Fig. 16, we apply local textures driven by the same reference image (left) to multiple diverse meshes. In each case, the style of the reference shirt is effectively applied to the mesh. Although these meshes have very different shapes, 3D PixBrush takes the context of the target mesh into account and properly structures the local texture in a way that makes sense on the given mesh.

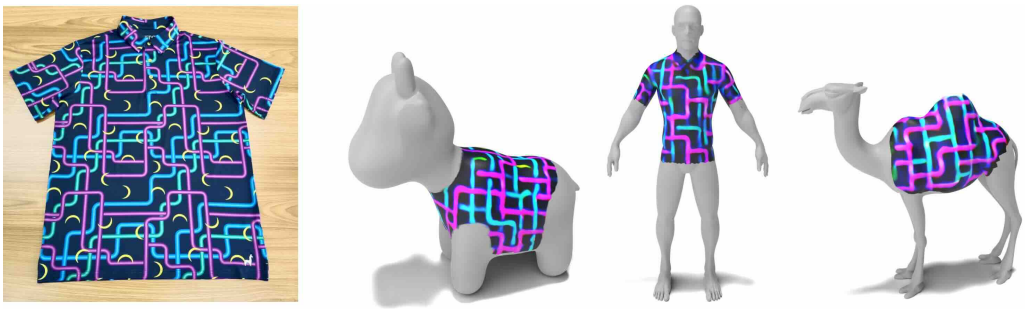

Figure 16: **Same Reference Image Applied to Diverse Meshes**. 3D PixBrush can produce local textures on a diverse set of meshes using the same reference image. Our method is capable of accurately contextualizing where the reference image ostensibly belongs on a variety of different input shapes.

Local Textures in Different Contexts. In Fig. 17, we apply a local texture driven by the same reference image (top left) in multiple different contexts. In each case, the style of the reference shirt is effectively applied to the mesh. Although these meshes have very different shapes, 3D PixBrush takes the context of the target mesh into account and properly structures the local texture in a way that makes sense on the given mesh geometry. For example, the first two meshes have a larger upper torso than the third mesh. On these two meshes, our method generates a larger shirt that is tight-fitting in this upper torso region, but is looser and, as a result, more wrinkled in the lower region. In contrast, the third mesh, which is more slender is given a smaller, tighter-fitting shirt that displays fewer wrinkles in the lower section.

Local Geometric Deformation. In Fig. 18, we combine our local image-driven texture edits with an off-the-shelf deformation framework Dinh et al. (2025) to achieve deformations in just the region

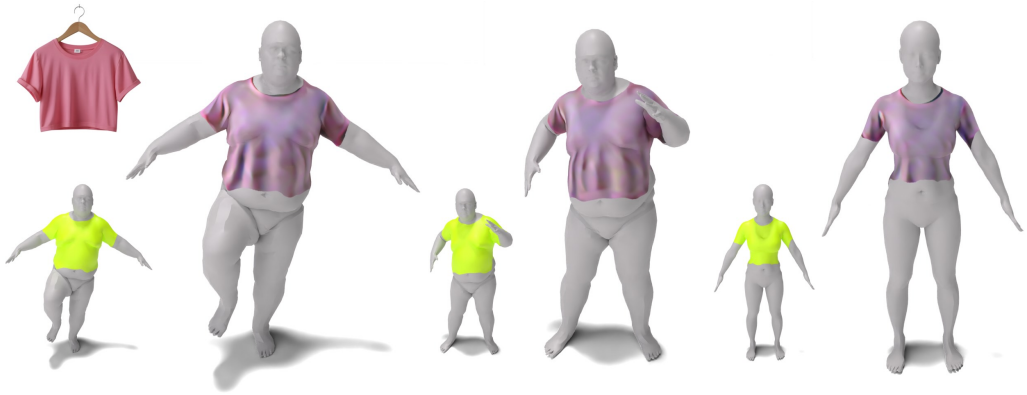

Figure 17: **Local Texture in Different Contexts.** Using a reference image of a pink crop top (top left), 3D PixBrush produces consistent localizations and local textures that capture the details of the shirt in the reference image and adapt it to fit the context across different poses of the same mesh or even different geometries altogether. This result highlights that this task requires a solution beyond a simple cut-and-paste of the reference image. We see that our method synthesizes wrinkles, texture details, and material changes to conform to different meshes.

of the local edit. Once we obtain our local edit, we use our learned localization mask to direct the deformation to the precise area of the local edit to obtain a local edit with both texture and geometric components.

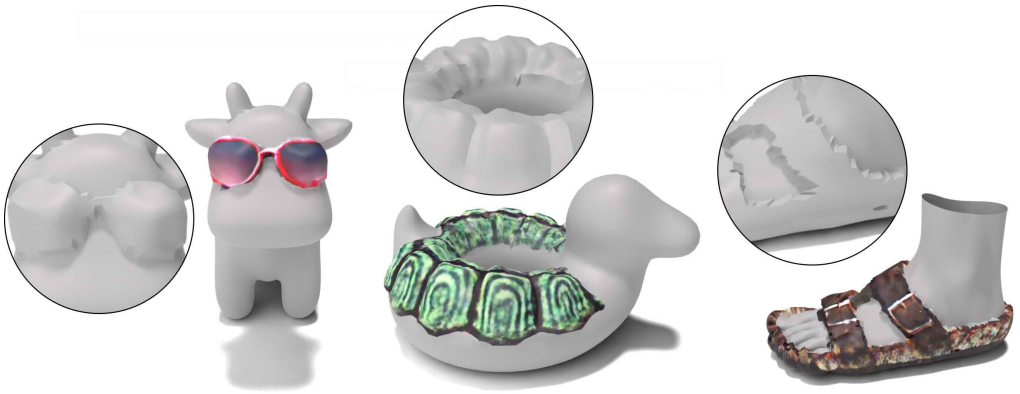

Figure 18: **Local Deformation Application.** Once we have obtained our local image-driven texture edits, we can apply an off-the-shelf mesh deformation framework Dinh et al. (2025) only in the region of our localization mask to obtain a local edit with both texture and geometric components.

Result on Genus One Mesh. In Fig. 19, we use the reference image of a turtle shell to learn a local texture on this rubber duck using our method. Even though the mesh is genus one and turtle shells are typically seen on flat, continuous surfaces, our method is able to adapt the local texture to the unique shape of the mesh. 3D PixBrush produces a local texture that wraps around the hole on the mesh's back while still displaying a pattern that matches the shell in the reference image.

Figure 19: **Result on Genus One Mesh.** We show the result of our approach using the turtle shell reference image (top right) to produce a local texture on this genus one rubber duck mesh. Even though turtle shells are typically seen on flat / genus zero surfaces, 3D PixBrush is able to produce a plausible shell texture that conforms to the unique shape of this mesh by properly wrapping around the hole on its back.

Figure 20: **Multi-View Results.** We show results of our method from multiple views. 3D PixBrush generates 3D consistent local textures that look plausible from arbitrary angles.