Applying Textures to Untextured Images Using Neural Networks

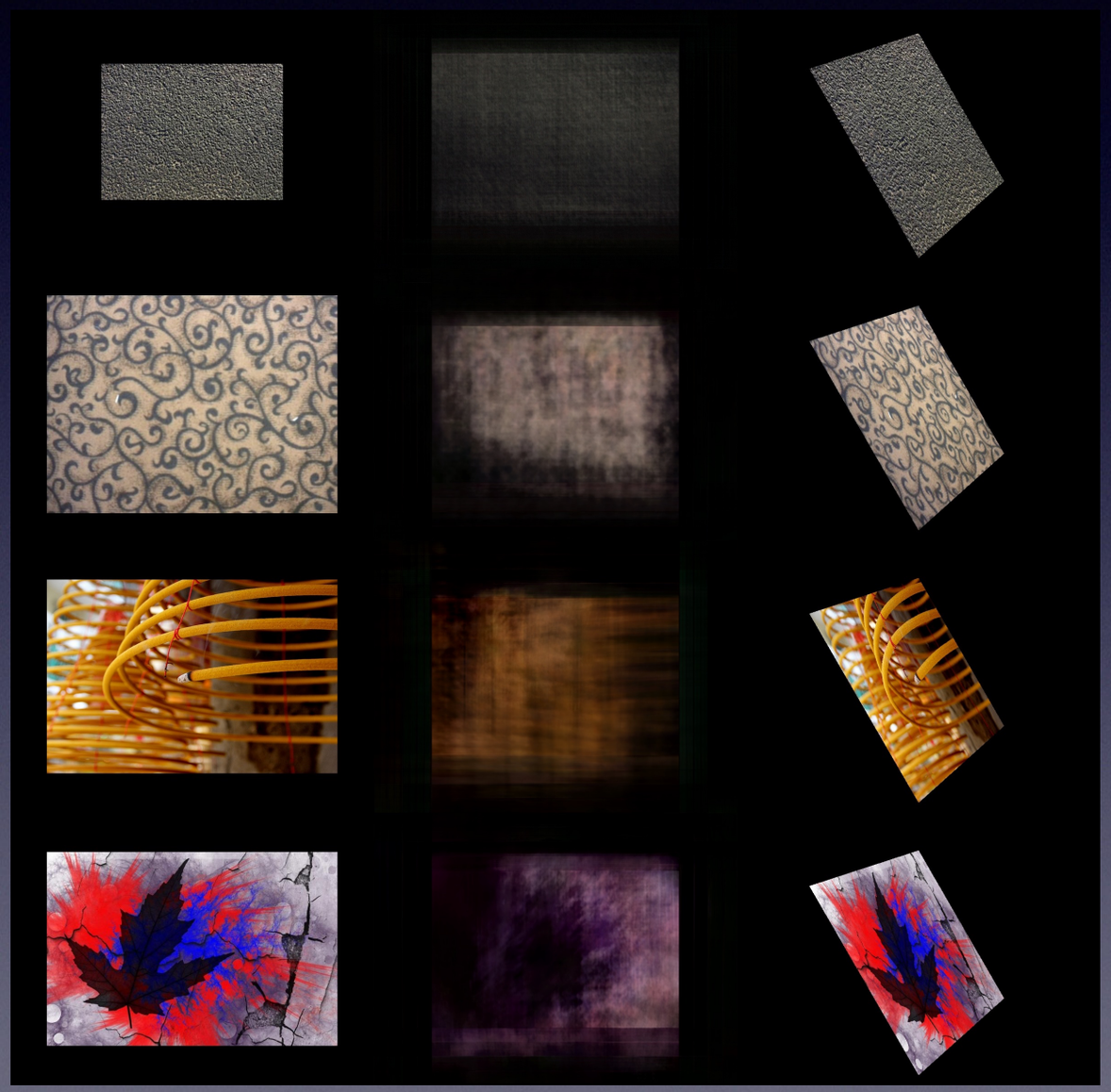
Stefan Palombo, Phillip Kuznetsov, Gabriel Gardner

Summary

- Our goal is to apply textures automatically to non-textured images of simple 3D renderings using fully-convolutional neural networks
- An effective way to automatically apply textures to textureless images could significantly reduce labor for projects that require handmade textures and also revitalize older sources with low-definition textures

Results: Rotated Plane

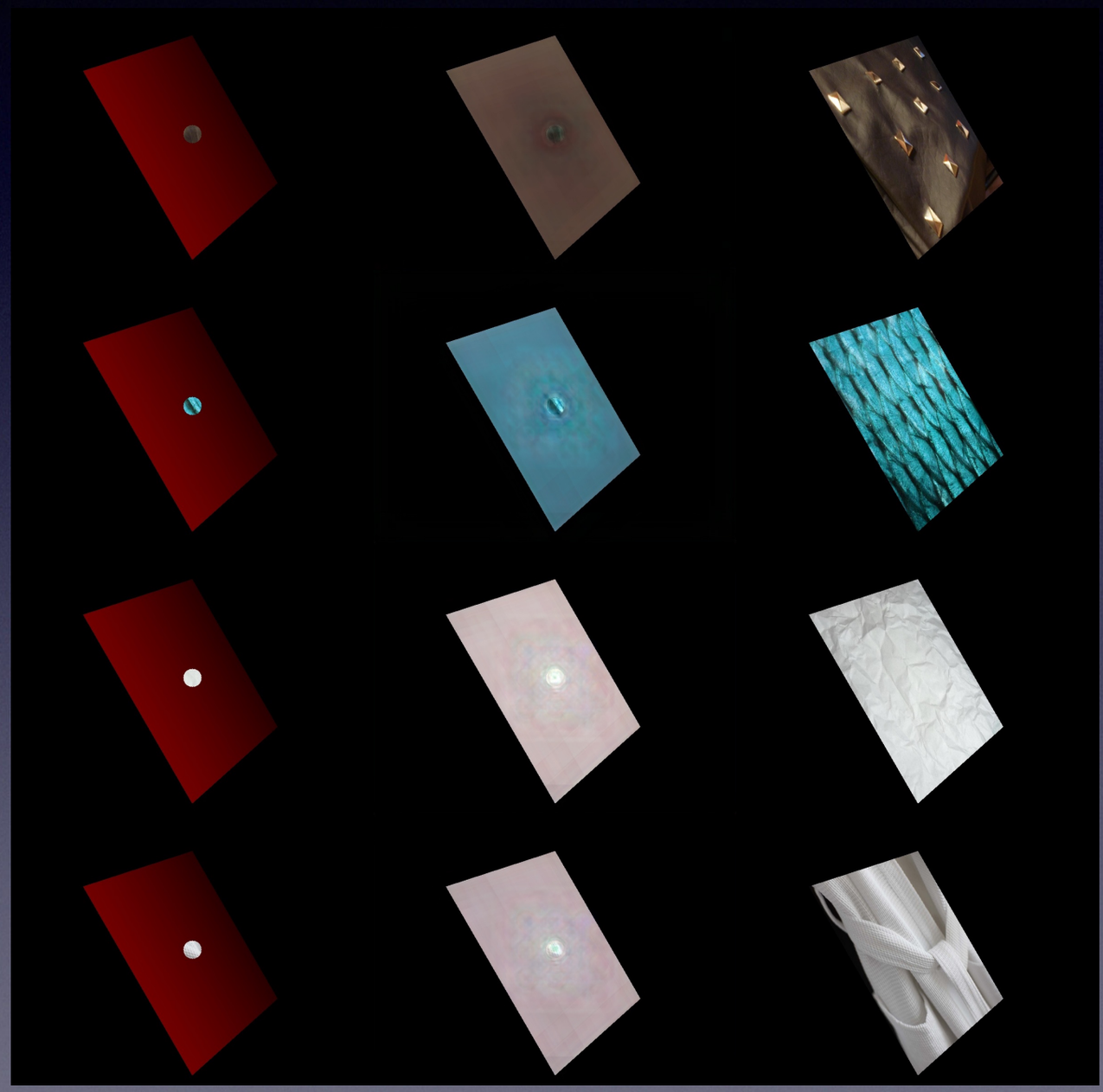
- We attempted to train a network to apply textures to a rotated plane purely based on a flat input texture
- This method clearly failed, but we did learn that neural nets need more explicit parameters to effectively learn geometric transformations



Input (left), Output (middle), Ideal Output (right)

Results: Texture Cue

- We then attempted to train networks on a rotated plane with a small texture cue that the network can then learn to apply to the image
- It manages to copy the proper color but not the texture
- We expect experimenting with the L2 loss function could help improve these results



Input (left), Output (middle), Ideal Output (right)

Results: L0 Smoothing

- In order to apply textures to real images with most of their textures removed, we created a dataset that uses the L0 norm to erase high frequency textures from an image

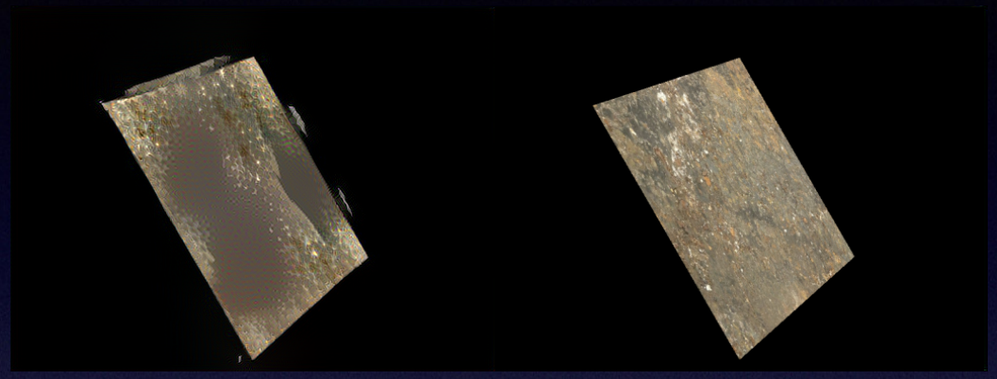
- We then train the networks to apply the textures back to the “textureless” L0 images

- We find these results fascinating but fear that this problem may be too complex for this project and may not continue down this path

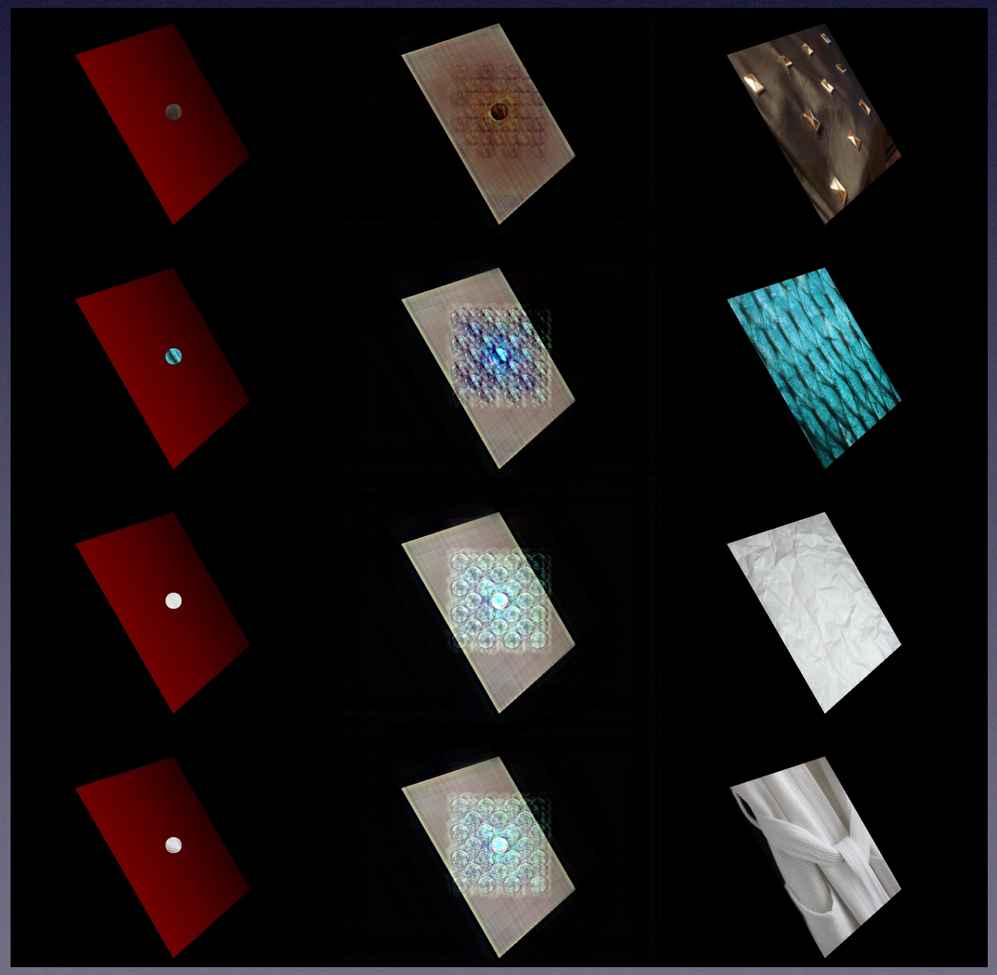
Input (left), Output (middle), Ideal Output (right)

Next Steps

- We have performed a baseline test with style transfer and believe there is potential to use style loss as part of our metric (results on top right)
- We have just started training a network with style loss and find the results (bottom right) intriguing
- Using Generative Adversarial Networks could likely improve our results



Output (left), Input (right)



Input (left), Output (middle), Ideal Output (right)