Interpolated SelectionConv for Spherical Images and Surfaces Appendix

A. Barycentric Interpolation Derivation

For a point p lying inside a square kernel within the neighborhood of 9 weights, there are 8 possible triangles that it may be contained within. Each triangle, however, is a right triangle with two bases with length d. Thus, using simple substitution, we can solve for a single triangle and apply the results to any of the 8. We represent our triangle with the three vertices $(0, 0)$, $(d, 0)$ and (d, d), while representing the transformed point as the coordinate (u,v). To convert any point p (relative to center of the kernel) to this coordinate system, we first take the $|p|$ so that it lies within the first quadrant, then given that it must be the case that $u \geq v$, we get the following substitution:

$$u = \max(|p|) \tag{1}$$

$$v = \min(|p|) \tag{2}$$

This simple substitutions works for points in any of the 8 original triangles, as shown in Fig. 1.

Finding the weights in barycentric interpolation can be expressed as the solving of a system of linear equations. Specifically,

$$\begin{bmatrix} x_0 & x_a & x_b \\ y_0 & y_a & y_b \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} w_0 \\ w_a \\ w_b \end{bmatrix} = \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \tag{3}$$

where w_n are the desired interpolation weights we are solving for. This system becomes drastically simplified when we substitute the fixed triangle and our substituted coordinates. Our system becomes

$$\begin{bmatrix} 0 & d & d \\ 0 & 0 & d \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} w_0 \\ w_a \\ w_b \end{bmatrix} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{4}$$

The inverse of this matrix is:

$$\begin{bmatrix} -1/d & 0 & 1 \\ 1/d & -1/d & 0 \\ 0 & 1/d & 0 \end{bmatrix} \tag{5}$$

Multiplying this out to solve the system gives us the following:

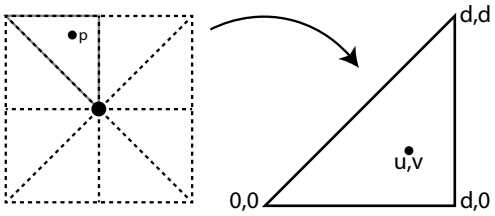

Figure 1. Regardless of which triangle a point p lands within kernel space, it can be substituted for a simple x-axis aligned isosceles right triangle.

$$\begin{array}{rcccc} w_0 & = & -u/d & & +1 \\ w_a & = & u/d & -v/d & \\ w_b & = & & v/d & \end{array} \tag{6}$$

Substituting the original point back in, we get our final result:

$$w_0 = 1 - \frac{\max(|p|)}{d} \tag{7}$$

$$w_a = \frac{\max(|p|) - \min(|p|)}{d} \tag{8}$$

$$w_b = \frac{\min(|p|)}{d} \tag{9}$$

B. Spherical Style Transfer Results

We provide additional results similar to those shown in Figure 5 in the paper, where we compare our approach to previous spherical stylization methods. Those results are shown in Fig. 2 - 5.

C. Mesh Stylization Results

We provide expanded versions of Figures 6 and 7 from the paper (Fig. 6 and Fig. 7) as well as additional results in Fig. 8 and Fig. 9.

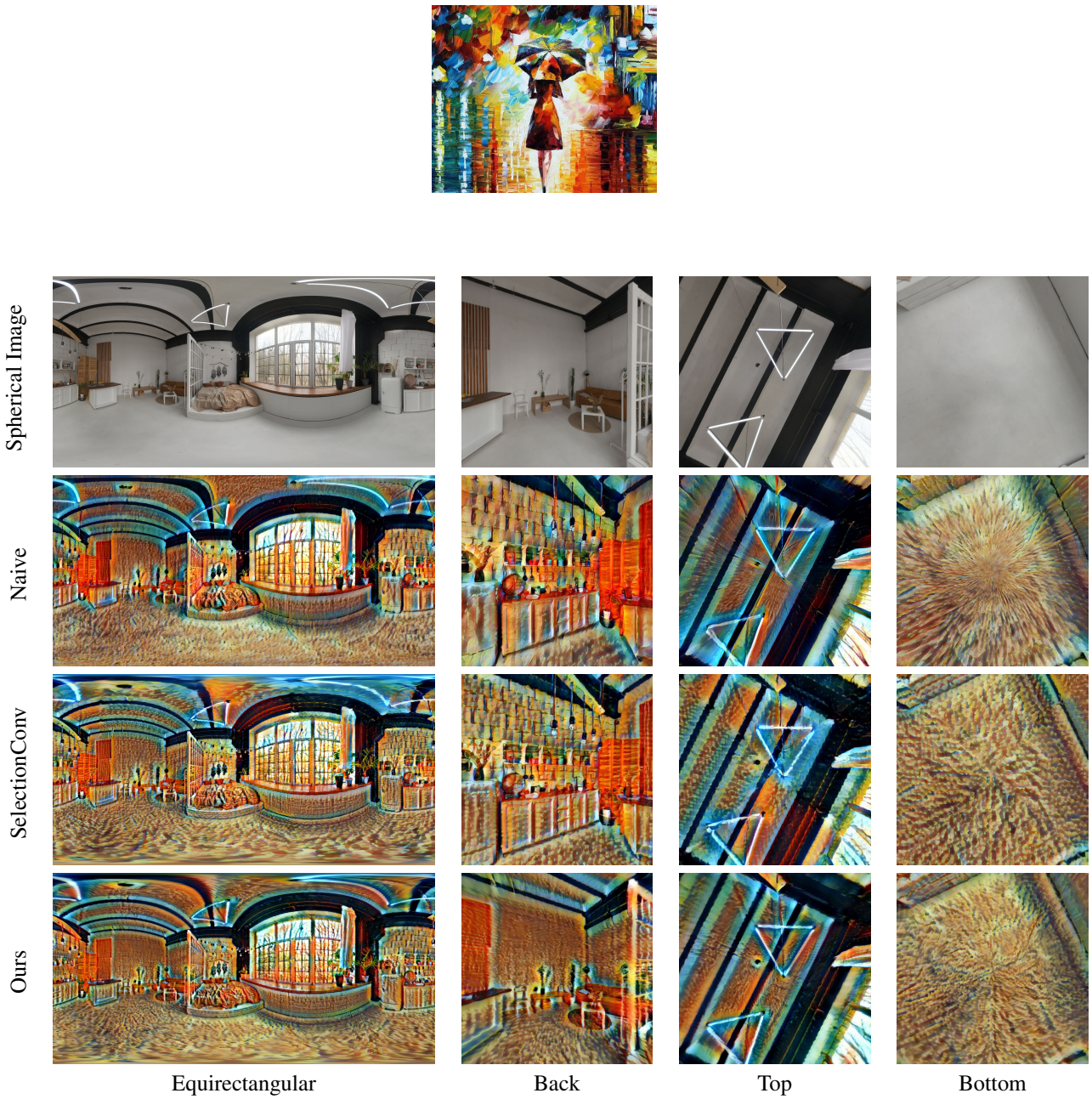

Figure 2. A 360° image (1st row), its stylization when naively stylizing the equirectangular image (2nd row), using the cube-map graph setup from the original SelectionConv paper (3rd row), and compared to our interpolated spherical representation (4th row). The equirectangular projection along with various views of the scene are presented. In the naive approach, note the vertical seam in the middle of the back view as well as the distortion in the top and bottom views. In the original SelectionConv results, note the artifacts in the top and bottom views along the seam connections (making an x shape). Those artifacts are removed with our new method. Public domain image courtesy of polyhaven.com.

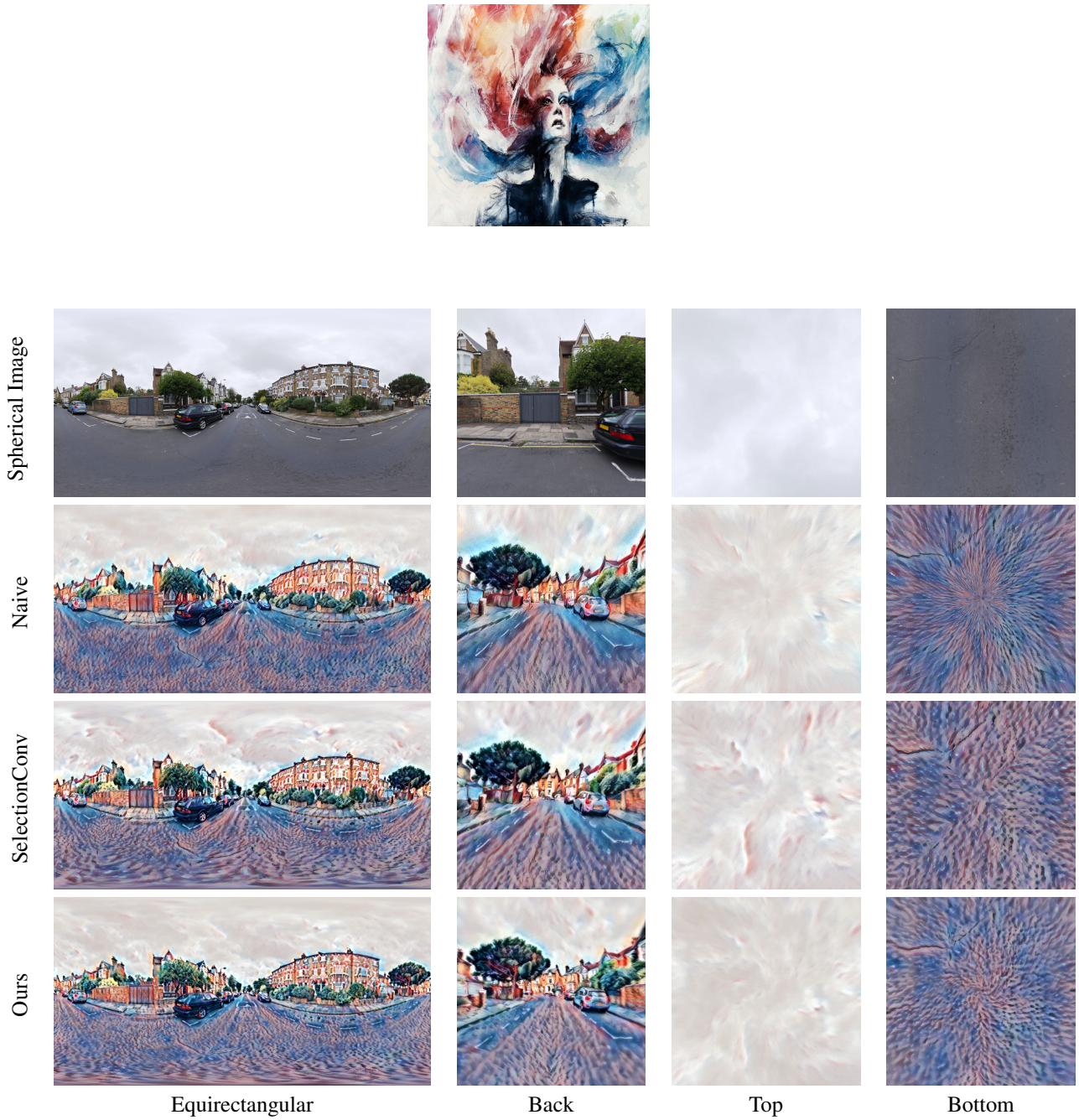

Figure 3. A 360° image (1st row), its stylization when naively stylizing the equirectangular image (2nd row), using the cube-map graph setup from the original SelectionConv paper (3rd row), and compared to our interpolated spherical representation (4th row). The equirectangular projection along with various views of the scene are presented. In the naive approach, note the vertical seam in the middle of the back view as well as the distortion in the top and bottom views. In the original SelectionConv results, note the artifacts in the top and bottom views along the seam connections (making an x shape). Those artifacts are removed with our new method. Public domain image courtesy of polyhaven.com.

Figure 4. A 360° image (1st row), its stylization when naively stylizing the equirectangular image (2nd row), using the cube-map graph setup from the original SelectionConv paper (3rd row), and compared to our interpolated spherical representation (4th row). The equirectangular projection along with various views of the scene are presented. In the naive approach, note the vertical seam in the middle of the back view as well as the distortion in the top and bottom views. In the original SelectionConv results, note the artifacts in the top and bottom views along the seam connections (making an x shape). Those artifacts are removed with our new method. Public domain image courtesy of polyhaven.com.

Figure 5. A 360° image (1st row), its stylization when naively stylizing the equirectangular image (2nd row), using the cube-map graph setup from the original SelectionConv paper (3rd row), and compared to our interpolated spherical representation (4th row). The equirectangular projection along with various views of the scene are presented. In the naive approach, note the vertical seam in the middle of the back view as well as the distortion in the top and bottom views. In the original SelectionConv results, note the artifacts in the top and bottom views along the seam connections (making an x shape). Those artifacts are removed with our new method. Public domain image courtesy of polyhaven.com.

Figure 6. When the texture map of the original mesh (a) is stylized naively (b), many artifacts are present along the UV seams. Stylizing with SelectionConv (c) removed some of those artifacts, but inconsistencies remain. Interpolated SelectionConv (d) retains far greater consistency along seams.

Figure 7. Example stylizations of high quality meshes with 4K textures (a). Naively stylizing the texture map (b) is slow, leaves artifacts on seams, and provides little control for the level of detail for the stylization. Our approach (c) removes seam artifacts while allowing the user to control the number of sampling points, which in turn determine the speed and detail of the stylization. Public domain meshes courtesy of polyhaven.com.

Figure 8. Given a style image (a) and 3D mesh (b), the naive method must style the texture map directly. It may do so on a downsampled version of the texture map (c) or at the original resolution (e), but both fail to properly handle UV seams or provide the user with an effective way to control the level of detail in the stylization. In comparison, our approach can control the number sampling points, with a low sampling giving a more smooth and larger-feature stylization (d) and a higher sampling giving a more varied stylization (f). Public domain meshes courtesy of polyhaven.com.

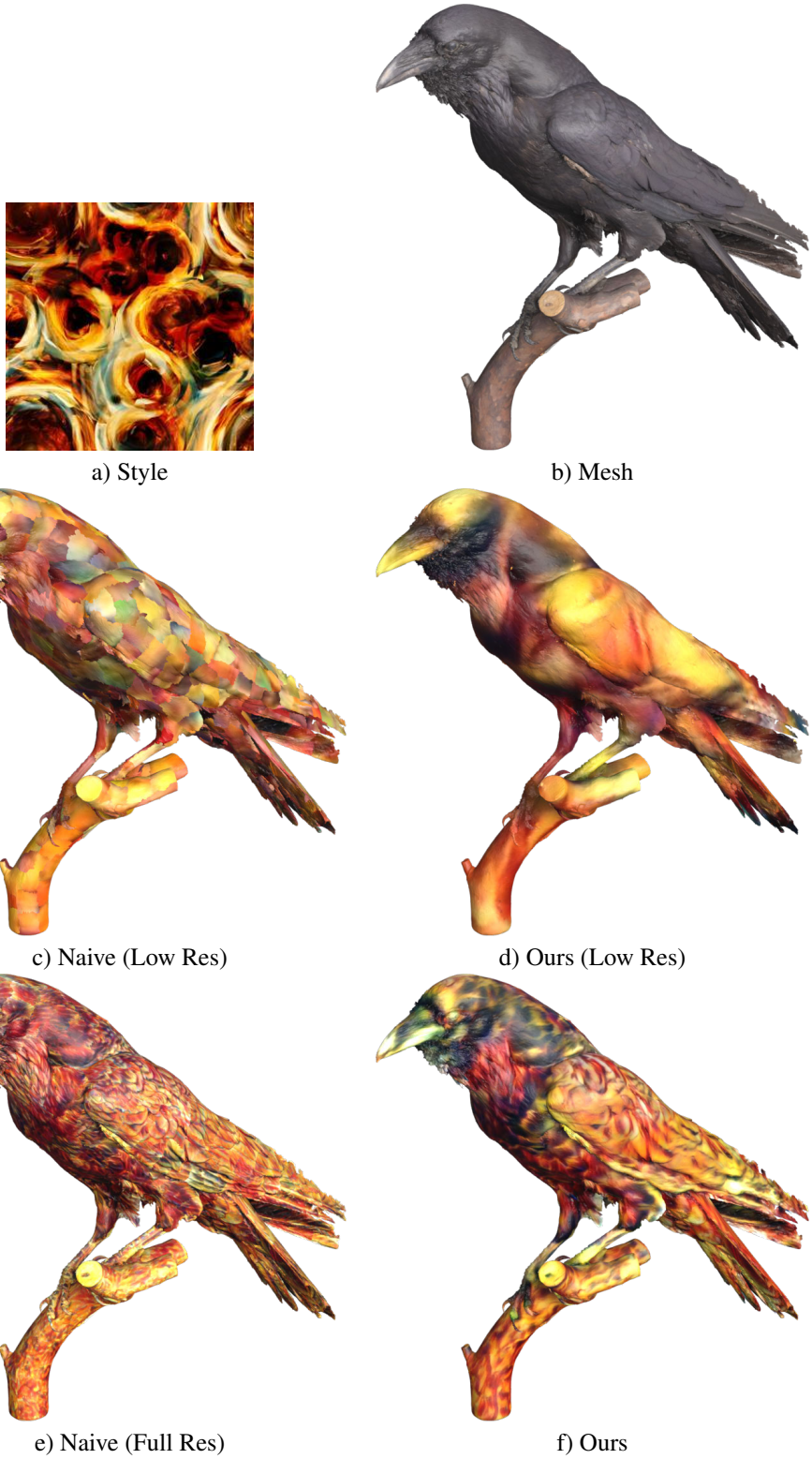

Figure 9. Given a style image (a) and 3D mesh (b), the naive method must style the texture map directly. It may do so on a downsampled version of the texture map (c) or at the original resolution (e), but both fail to properly handle UV seams or provide the user with an effective way to control the level of detail in the stylization. In comparison, our approach can control the number sampling points, with a low sampling giving a more smooth and larger-feature stylization (d) and a higher sampling giving a more varied stylization (f).