1. Learned Edge Priors

The learned edge priors described in Section 3.2 of the manuscript are presented here. Since the pixel coordinates are integers, the distance between two pixels can only be countable discrete values (i.e., \( t = \{0, 1, \sqrt{2}, 2, \sqrt{5}, 2\sqrt{2}, \ldots \} \)). The learned priors are computed from 2,184 upright frontal face images cropped from the CMU Multi-PIE dataset [6] with resolution of 320 \times 240 pixels. Some training images are shown in Figure 1. To learn the statistical priors, we downsample the training images to low-resolution images through Eq. 3, where the kernel width of \( G \) is set to 1.6 and the scaling factor is set as 4. The parameters are the same as those used in [9] for fair comparison.

The learning process described in Section 3.2 on Lines 376 to 427 of the manuscript generates a nonparametric distribution via a lookup table that maps features \((m_p, m_c, t)\) to \(\tilde{m}'_p\) based on gradient magnitudes. Figure 2 displays the lookup table values in different feature \(t\). In Figure 2(a), when \(t\) is zero, pixels \(p\) and \(r\) are identical as well as \(m_p\) and \(m_c\). Thus we simplify the two-dimension source domain of \((m_p, m_c)\) to one-dimension domain of \(m_p\) only. The averaged \(\tilde{m}'_p\) values are shown as the red circles while the standard deviation in each bin is shown as the blue crosses. Note that the \(\tilde{m}'_p\) values do not monotonically increase because the types of edges in the training images (Figure 1) are limited, e.g., the collars, the shadow of nose, face contours, the boundary of hairs, etc. Figure 2(b)-(h) show the lookup table values for other feature value \(t\). For most collected samples \((m_p, m_c, t)\), the \(m_c\) values are greater than \(m_p\) values. This is what is expected as pixel \(c\) is at the center of an edge, whose magnitude of gradients is likely greater than off-center pixel \(p\). As illustrated in Figure 2(b)-(h), there are more non-empty bins (where \(\tilde{m}'_p\) values exist) whose \(m_c\) values are greater than the \(m_p\) values. When \(m_c\) is greater, \(\tilde{m}'_p\) is likely greater because a sharp edge in low-resolution is likely generated by a sharp edge in high-resolution. When distances \(t\) is smaller, the correlation between \(m_c\) and \(\tilde{m}'_p\) is stronger, which reflects the effect of distance on edge sharpness in the training images.

![Figure 1: Training images from the CMU Multi-PIE dataset.](image-url)
Figure 2: Learned edge priors (graphics best viewed on a color display).
Figure 2: Learned edge priors (graphics best viewed on a color display).
2. More Experimental Results

We show more experimental results here to demonstrate the effectiveness of the proposed algorithm for various test images. Figure 3-8 are from the CMU Multi-PIE dataset [6] as well as Figure 9-10 from the PubFig dataset [8]. The proposed algorithm preserves the consistency of component details such as the teeth in Figure 3(f) and Figure 6(f), which show better quality than the ones generated by existing algorithms in Figure 3(b)(c)(d)(e) and Figure 6(b)(c)(d)(e). Although the transferred component details are different from the ground truth in Figure 3(g) and Figure 6(g) (which is unlikely to be recovered by any super-resolution algorithm), the results are visually pleasing and correct. Likewise, other facial components are well generated by the proposed method, e.g., lips in Figure 7, noses in Figure 5 and Figure 8, eyes in Figure 4 and Figure 9, and eyebrows in Figure 10.

![Figure 3](image1.png)

Figure 3: Qualitative comparison for 4 times upsampled upright frontal faces (results best viewed on a high-resolution display). The test image is from the CMU Multi-PIE dataset, where the subjects of training and test sets do not overlap.

![Figure 4](image2.png)

Figure 4: Qualitative comparison for 4 times upsampled upright frontal faces (results best viewed on a high-resolution display). The test image is from the CMU Multi-PIE dataset, where the subjects of training and test sets do not overlap.
Figure 5: Qualitative comparison for 4 times upsampled upright frontal faces (results best viewed on a high-resolution display). The test image is from the CMU Multi-PIE dataset, where the subjects of training and test sets do not overlap.

Figure 6: Qualitative comparison for 4 times upsampled upright frontal faces (results best viewed on a high-resolution display). The test image is from the CMU Multi-PIE dataset, where the subjects of training and test sets do not overlap.

Figure 7: Qualitative comparison for 4 times upsampled non-frontal faces (results best viewed on a high-resolution display). The test image is from the CMU Multi-PIE dataset, where the subjects of training and test sets do not overlap.
Figure 8: Qualitative comparison for 4 times upsampled non-frontal faces (results best viewed on a high-resolution display). The test image is from the CMU Multi-PIE dataset, where the subjects of training and test sets do not overlap.

Figure 9: Qualitative comparison for 4 times upsampled upright frontal faces (results best viewed on a high-resolution display). The test image is from the PubFig dataset.

Figure 10: Qualitative comparison for 4 times upsampled upright frontal faces (results best viewed on a high-resolution display). The test image is from the PubFig dataset.
3. Failure Case

We show two failure cases here with explanations. The proposed algorithm does not generate high-quality images if the intermediate image $I_b$ described on Lines 230 and 264 of the manuscript is incorrectly labeled with the landmark points. While the adopted algorithm [17] generally performs well, it fails to correctly label landmark points in some cases. An example is shown in Figure 11(a) where the landmark points of the left eye are incorrectly labeled so that their positions are too high. As a result, although high-frequency details of an eye are consistently transferred, the size and the position of the transferred eye are incorrect, as shown in 11(f). We note that the other state-of-the-art algorithms do not generate high-quality images in this example.

Another reason causing failures is the ambiguity problem, i.e., similar features in low-resolution do not match to similar features in high-resolution due to the downsampling process. Figure 12 shows an example where the determined exemplar image is very similar in low-resolution but quite different in high-resolution. As a result, the high-frequency details of a widely open eye is transferred as shown in Figure 12(f) but a narrow eye will be better. We note that the other state-of-the-art algorithms do not generate high-quality images in this example.

![Image of failure case](image1.jpg)

**Figure 11:** A failure case of the proposed algorithm caused by wrongly labeled landmark points (results best viewed on a high-resolution display). Note that the region bounded by the landmark points in (a) is used to generate high-frequency details in (f). Since the landmarks are not correctly detected, the results in (f) contain noticeable artifacts.

![Image of another failure case](image2.jpg)

**Figure 12:** A failure case of the proposed algorithm caused by the ambiguity problem (results best viewed on a high-resolution display).
4. Algorithms

Algorithm 1 shows the process to solve Eq. 6 of the manuscript

$$I_d = \arg \min_i \| (I \otimes G) \downarrow - I_i \|^2 + \beta \sum_{k=1}^K \| f_k(I) - T_k \|^2.$$ 

The optimization problem generates a high-resolution image whose directional similarity maps are similar to $T_k$ as well as downsampled image is similar to $I_i$. An initial image is generated from $I_i$ through bicubic interpolation and then alternatively updated to minimize the total energy value. To minimize the first energy term, a descent direction is computed on Line 6, and a line search is executed on Lines 7 to 16 to find a new image with smaller energy value. To ensure the maps of directional similarity close to $T_k$, an update through weight average based on the maps of directional similarity is executed on Line 5. The comparison of the original energy value and new energy value on Line 10 ensures that the new image has a decreasing energy value. Thereafter, we update the image by the new image and start a new iteration from Line 3. The iterative minimization stops at a given iteration number.

Algorithm 1: Direction-Preserving Upsampling

**Data:** Low-resolution image $I_l$, scaling factor $s$, Gaussian kernel $G$, expected directional similarity maps $\{T_k\}$, parameter $\beta$, iteration number $l$, line search step number $m$

**Result:** High-resolution image $I_s$

1. $w_k(x, y) \leftarrow \frac{T_k(x, y)}{\sum_k T_k(x, y)}$ //Compute directional similarity weights
2. $I \leftarrow s$-time bicubic interpolated $I_l$ //Initialize
3. for $i \leftarrow 1$ to $l$ do
4.  $e \leftarrow \| (I \otimes G) \downarrow - I_i \|^2 + \beta \sum_{k=1}^K \| f_k(I) - T_k \|^2$ //Compute the original energy value
5.  $I_1(x, y) \leftarrow \sum_k w_k(x, y) \cdot I(x + \Delta x_k, y + \Delta y_k)$ //Generate a new image by weight average, where $\Delta x_k, \Delta y_k$ are the directional offsets
6.  $D \leftarrow ((I_1 \otimes G) \downarrow - I_l) \uparrow \otimes G$ //Compute the descent direction
7.  $\tau \leftarrow 0.2$ //Set initial step length
8.  for $j \leftarrow 1$ to $m$ do
9.      $I_2 \leftarrow I_1 - \tau \cdot D$ //Generate a new image
10.     if $\| (I_2 \otimes G) \downarrow - I_i \|^2 + \beta \sum_{k=1}^K \| f_k(I_2) - T_k \|^2 < e$ then
11.        $I \leftarrow I_2$ //Update the image
12.        break
13.     else
14.        $\tau \leftarrow 0.5 \tau$ //Update the step length
15.     end
16.  end
17. end
18. $I_d \leftarrow I$ //Return the result
Algorithm 2 shows the process to solve the optimization problem of Eq. 1 in our experiments

\[ I_h = \arg \min_I \| \nabla I - U \|_2^2 \quad \text{s.t.} \quad (I \otimes G) \downarrow = I_l. \]

Although the problem is theoretically solvable by a quadratic programming package, the number of variables (as many as the pixel numbers of the output image) in the constraint term makes it difficult to solve in practice. We relax the constraint term by changing it as a regularization term and computing a descent direction in

\[ I_h = \arg \min_I \beta \| \nabla I - U \|_2^2 + \| (I \otimes G) \downarrow - I_l \|_2^2. \]

Along the descent direction, all variables can be updated simultaneously so that the new optimization problem can be solved efficiently. We decrease the parameter \( \beta \) in each iteration on Line 6 to make a closer approximation to the original optimization problem. Since the given initial image \( I_n \) may not satisfy the constraint term \( \| (I \otimes G) \downarrow - I_l \| = 0 \), we apply back-projection on Lines 2 to 4 to reduce the residue. The original energy value \( e \) is computed on Line 8, and a descent direction for generating a new image is computed on Line 9, where the \( \text{Div}() \) is a divergence operator and \( \nabla_k \) computes the derivatives for direction \( k \). We execute a line search on Lines 10 to 14 and record the energy values of all step lengths in an array \( r \). We find the best step index \( j^* \) and check the energy value \( r[j^*] \) on Line 16. If the new energy value \( r[j^*] \) is smaller than the original energy value \( e \), the image is updated on Lines 17 to 18.

---

**Algorithm 2: Generating the Output Image**

**Data:** Low-resolution image \( I_l \), gradient maps \( \{U_k\} \), Gaussian kernel \( G \), initial image \( I_n \), tolerance value \( t \), update number \( n \), iteration number \( l \), line search step number \( m \)

**Result:** High-resolution image \( I_h \)

1. \( I \leftarrow I_n \) Initialize
2. while \( \| (I \otimes G) \downarrow - I_l \| > t \) do
3. \( I \leftarrow I - \left( \left( (I \otimes G) \downarrow - I_l \right) \uparrow \right) \otimes G \) //Use back-projection to reduce the residue of the initial image
4. end
5. for \( k \leftarrow 1 \) to \( n \) do
6. Set \( \beta \leftarrow 2^{1-k} \)
7. for \( i \leftarrow 1 \) to \( l \) do
8. \( e \leftarrow \beta \| \nabla I - U \|_2^2 + \| (I \otimes G) \downarrow - I_l \|_2^2 \) //Compute the original energy value
9. \( D \leftarrow \beta \left( \sum_{k=1}^{K} \nabla_k U_k - \text{Div}(I) \right) + \left( \left( (I \otimes G) \downarrow - I_l \right) \uparrow \right) \otimes G \) //Compute the descent direction
10. for \( j \leftarrow 1 \) to \( m \) do
11. \( \tau \leftarrow 2^{1-j} \) //Set the step length
12. \( I' \leftarrow I - \tau D \) //Compute a new image
13. \( r[j] \leftarrow \beta \| \nabla I' - U \|_2^2 + \| (I' \otimes G) \downarrow - I_l \|_2^2 \) //Record the new energy value in an array
14. end
15. \( j^* \leftarrow \text{argmin}_j r[j] \) //Find the index whose energy value is minimal
16. if \( r[j^*] < e \) then
17. \( \tau \leftarrow 2^{1-j^*} \) //Compute the step length for updating the image
18. \( I \leftarrow I - \tau D \) //Update the image
19. end
20. end
21. \( I_h \leftarrow I \) //Return