Geometric Transforms

R.Rohith
EP21B030

Since translation, rotation, and scaling are all affine transformations, I implemented a routine that takes an affine transformation matrix and an input image, applies the transformation, and performs bilinear interpolation during target-to-source mapping.

```python
def affine_transfrom(I, T, n_out, m_out):
    """
    Function to perform affine transformation T on I
    n_out, m_out : Desired shape of the output image
    """
    n, m = I.shape
    inv_T = np.linalg.inv(T)
    I_out = np.zeros((n_out,m_out))
    for x_ in range(n_out):
        for y_ in range(m_out):
            x, y, w = inv_T.dot(np.array([x_,y_,1]))
            x, y = x/w, y/w
            # bilinear interpolation using
            # multilinear polynomial fitting
            x_1, y_1 = int(x), int(y)
            x_2, y_2 = int(x) + 1, int(y) + 1
            if 0 <= x_1 < n and 0 <= x_2 < n and
                    0 <= y_1 < m and 0 <= y_2 < m:
                f = np.array([I[x_1,y_1], I[x_1,y_2],
                              I[x_2,y_1], I[x_2,y_2]])
                N = np.array([[1, x_1, y_1, x_1*y_1],
                             [1, x_1, y_2, x_1*y_2],
                             [1, x_2, y_1, x_2*y_1],
                             [1, x_2, y_2, x_2*y_2]])
                a = np.linalg.inv(N).dot(f)
                val = np.dot(a,np.array([1, x, y, x*y]))
            else:
                val = 0.0
            I_out[x_,y_] = val
    return I_out
```

The results are shown on the following pages.

1 Translation

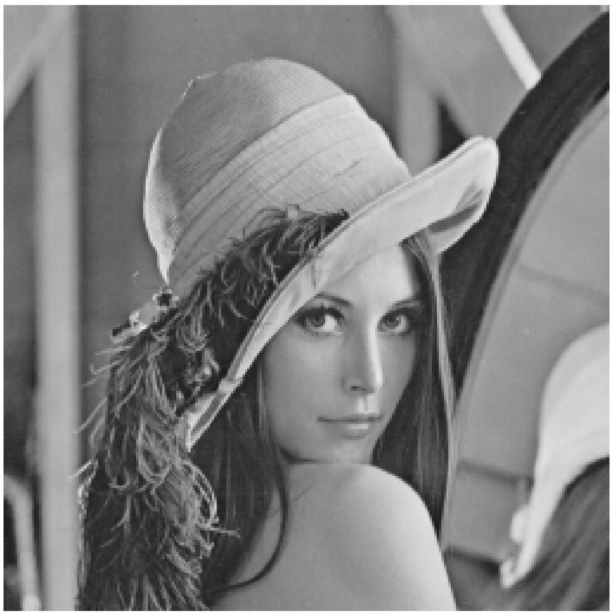
Figure 1: Source image

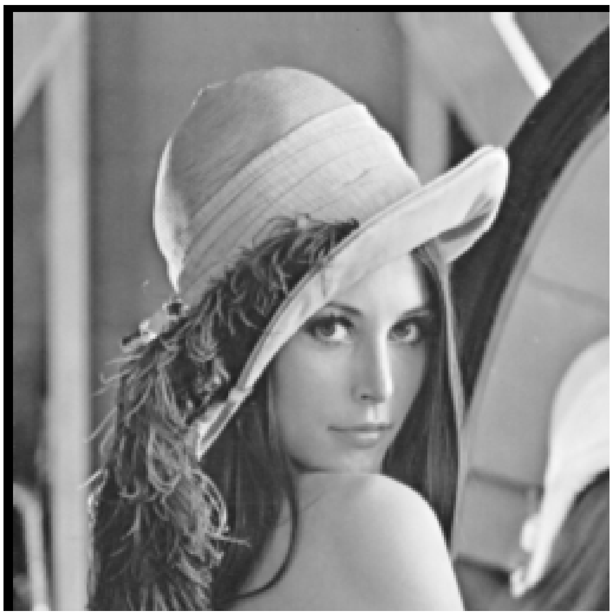
Figure 2: After translating by $t_x = 3.75$ and $t_y = 4.3$ pixels

The shape of the output image is the same as that of the input image.

2 Rotation

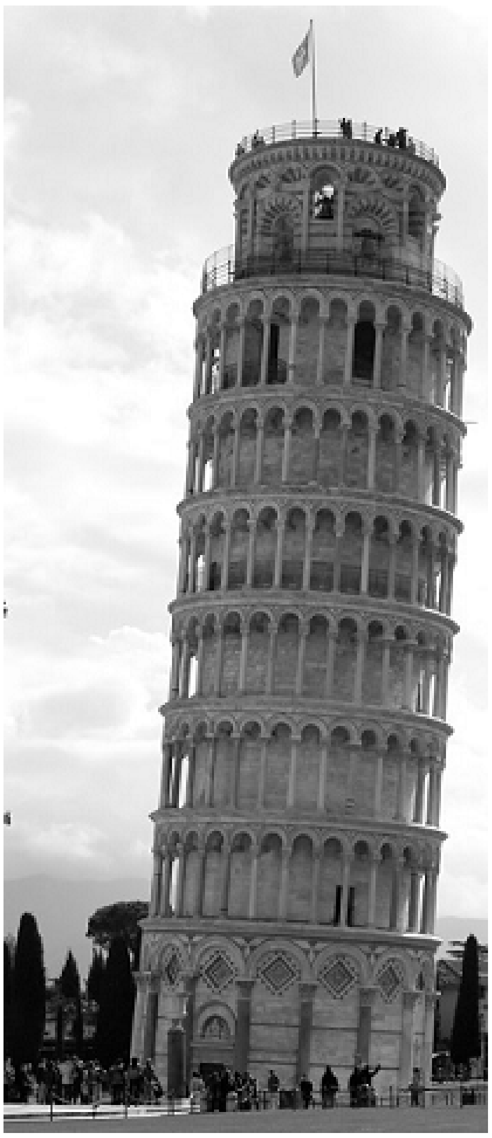

Figure 3: Source Image

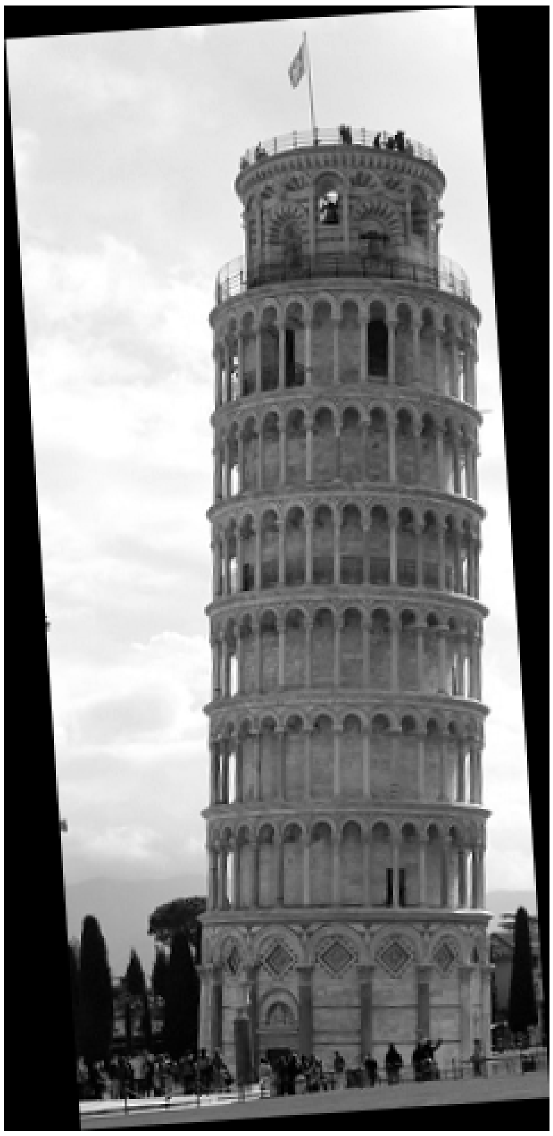

Figure 4: After rotating by 3.97°

Since the rotation happens around the axis passing the upper left corner of the image, I had to apply translation to bring the center of the source image to the center of the new target with an appropriate bounding box.

3 Scaling

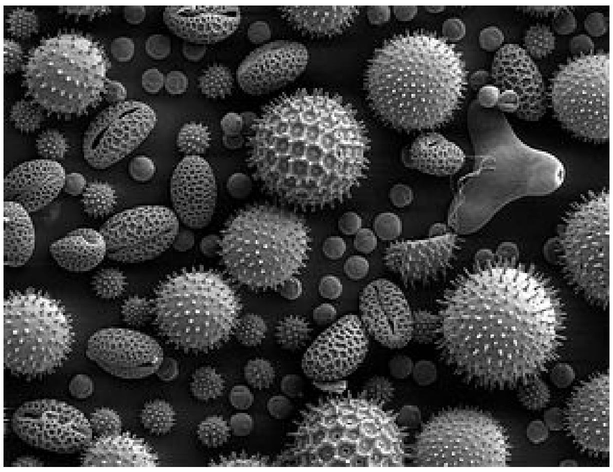
Figure 5: Source Image

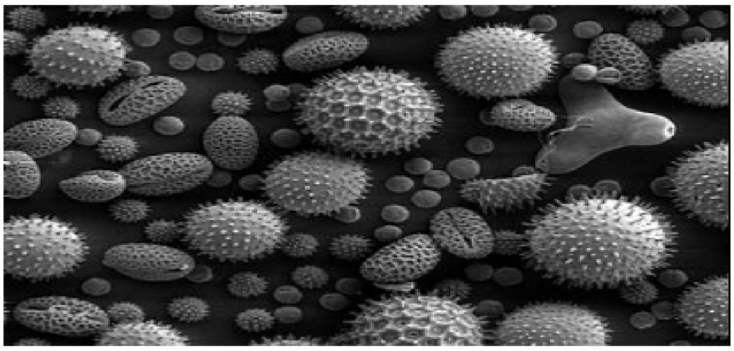
Figure 6: After scaling by factors of 0.8 and 1.3