2

1 Fourier-based Translation Approach

In the paper, a Fourier domain-based approach is presented to find the translation between two images. Let us assume that we have two images $f_1(x, y)$ and $f_2(x, y)$ with:

$$f_2(x, y) = f_1(x - x_0, y - y_0)$$

Because of the Fourier shift theorem, the Fourier transforms of these images will differ only by a phase factor:

$$\mathscr{F}_2(x', y') = \mathscr{F}_1(x', y')e^{-j2\pi(x'x_0)+y'y_0))}$$

To separate out the phase factor, we then calculate the cross-power spectrum as:

$$\frac{\mathscr{F}(x', y')\mathscr{F}(x', y')^*}{\|\mathscr{F})(x', y'\mathscr{F}(x', y')\|} = e^{j2\pi(x'x_0+y'y_0)}$$

The IFT of the above function is simply $\delta(x - x_0, y - y_0)$. Plotting this function will provide us with a clear spike at the displacement (x_0, y_0). Thus we can find the displacement between the two images that have a translation.

1.1 Time Complexity of this Approach

This procedure involves the calculation of two FFTs for image $f_1(x, y)$ and $f_2(x, y)$. And one IFFT of the cross-power spectrum. The time complexity of a single FFT/IFFT of an $N \times N$ image is:

$$T = O(N^2 \log(N^2)) = O(N^2 \log(N))$$

Thus, the time complexity of this procedure is:

$$T = O(3N^2 \log(N)) = O(N^2 \log(N))$$

1.2 Comparison of Complexity with Pixel-based Methods

We will compare this to the MSSD method that we studied in the early weeks of this course. In the procedure, we iterate the translation parameters t_x and t_y over a search range(say M_1 for t_x and M_2 for t_y). For every such pair, we have a Transformation matrix and we transform image 2 using it. Finally, we calculate the MSSD of these two images and pick the transformation matrix that corresponds to the least MSSD value. Here, we are using the brute-force method. The time complexity of Transforming an $N \times N$ image is $O(N^2)$ The time complexity of MSSD calculation is $O(N^2)$. Thus total complexity of the procedure is $O(M_1 * M_2 * N^2 * N^2) = O(M_1 M_2 N^4)$. We can clearly see that the Fourier-based method is way faster and more precise.

2 Correcting for Rotation

In order to find the rotation between two images, let us consider images $f_1(x, y$ and $f_2(x, y)$ with:

$$f_2(x, y) = f_1(x\cos(\theta_0) + y\sin(\theta_0) - x_0, -x\sin(\theta_0) + y\cos(\theta_0) - y_0)$$

The Fourier transform of these images is given by:

$$\mathscr{F}_2(x', y') = \mathscr{F}_1(x'\cos(\theta_0) + y'\sin(\theta_0), -x'\sin(\theta_0) + y'\cos(\theta_0))e^{-j2\pi(x'x_0)+y'y_0))}$$

Now, if we only look at the magnitudes of $\mathscr{F}_1$ and $\mathscr{F}_2$, let them be M_1 and M_2:

$$M_2(x', y') = M_1(x'\cos(\theta_0) + y'\sin(\theta_0), -x'\sin(\theta_0) + y'\cos(\theta_0))$$

Hence, one is the rotated replica of the other. If we represent this magnitude in polar coordinates, then we can see this rotation as a translation in θ, ie.:

$$M_1(\rho, \theta) = M_2(\rho, \theta - \theta_0)$$

Hence, we can apply the phase correlation procedure elaborated in section 1 to find out this translation in θ and make a correction for rotation.
If the two images have different scaling, their Fourier magnitude will have the following relation:

$$M_1(\rho, \theta) = M_2(\rho/a, \theta - \theta_0)$$

Hence, we can still employ phase correlation technique to find θ_0