APPLICATION AND OPTIMIZATION OF CROSS-CORRELATION ALGORITHM FOR IMAGE REGISTRATION

Baglaeva E.A., Tsapko S.G, Shepetovsky D.V.
Scientific advisor: Tsapko S.G.
National Research Tomsk Polytechnic University
634050, Tomsk, Lenin Avenue, 30
E-mail: eab14@tpu.ru

Introduction

Image registration is the process of transforming different sets of data into a single coordinate system. The data can be multiple photographs, data from different sensors, times, depths, or viewpoints. Image registration is used in computer vision, medical imaging, biological imaging and brain mapping, military automatic target recognition, and satellite images compiling and analysis. This process is necessary to perform comparison or integration of data obtained from these different measurements. [1]

The process of image registration consists of four steps:

1. Feature detection.
2. Feature matching.
3. Transform model estimation.
4. Image resampling and transformation.

The first step, feature detection, means finding correspondence between the image and the template. This work will consider application a cross-correlation algorithm to this first step.

Cross-correlation algorithm

In statistics, **dependence** is any statistical relationship between two random variables or two sets of data. **Correlation** refers to any of a broad class of statistical relationships involving dependence.

The most familiar measure of dependence between two quantities is the **Pearson product-moment correlation coefficient**, commonly called simply "the correlation coefficient". It is obtained by dividing the covariance of the two variables by the product of their standard deviations. [2]

Pearson's correlation coefficient when applied to a sample is commonly represented by the letter *r* and may be referred to as the sample correlation coefficient or the sample Pearson correlation coefficient. We can obtain a formula for *r* by dividing estimates of the covariances to estimates of the variances of the series. That formula for *r* is: [3]

$$r=\frac{\sum_{i=1}^{n}(X_i-\overline{X})(Y_i-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\overline{Y})^2}}$$

Cross correlation is a standard method of estimating the degree to which two series are correlated.

Let us consider two series *x*(*i*) and *y*(*i*) where *i*=0,1,2...N-1. The cross correlation *r* at delay *d* is defined as:

$$r(d)=\frac{\sum_i (x[i]-mx)(\mathrm{y}[i-d]-my)}{\sqrt{\sum_i (x[i]-mx)^2}\sqrt{\sum_i (\mathrm{y}[i-d]-my)^2}}$$

Where *mx* and *my* are the means of the corresponding series. If the above is computed for all delays *d*=0,1,2,...*N*-1, it results in a cross correlation series of twice the length as the original series.

It is necessary to determine cases when the difference between the index *i* and the delay *d* is out of range 0, 1, 2, … *N*-1. There are several ways for solving this problem. First, it is possible to set the value *y*(*i*-*d*) as 0 for *i*<*d* and *i*>*N*+*d*-1. Second, the indexes can be "wrapped" back within range, i.e. *x*(-1) = *x*(*N*-1), *x*(*N*+5) = *x*(5), etc. Third, these values can be ignored and not taken into observation. Second option is applied if the series is assumed to be circular, otherwise third option is applied.

The value of Pearson's correlation coefficient is between +1 and −1.

However, limited length of data types and the use of division operator result in a noticeable computational error if this formula is used.

Therefore, the denominator can be discarded, leaving only the numerator. The obtained formula represents the un-normalized cross-correlation coefficient.

The formula above is valid for the analysis of one-dimensional arrays. The same method applied to image processing is called 2D cross-correlation. In this method, image is represented as a numerical matrix, storing the values of different channels: brightness, hue, saturation, etc.

$$r[i][j]=\sum_{jj=0}^{jj<N_j}\sum_{ii=0}^{ii<N_i}(mask[ii+N_i][\mathrm{jj}+N_j]-$$
$$-\overline{mask})(image[i+ii][\mathrm{j}+jj]-\overline{image})$$

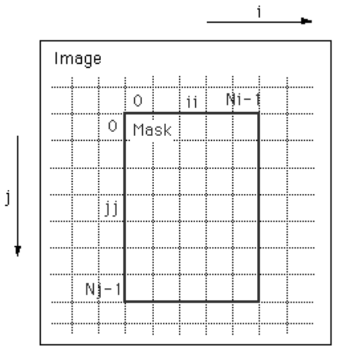

Fig. 1. The image and the mask

Let us consider the image and the mask (Fig. 1). The upper left corner of the mask (with coordinates (0, 0)) corresponds to a currently analyzed pixel in the image. The mask is moving over the image and as the cross correlation is calculated, this process forms a 2D array of correlation coefficients. [4]

It should be noted that the mean of the mask is a constant value, while the mean of the image has to be recalculated at every pixel of the image.

Optimization of the algorithm

The complexity of calculating a cross-correlation coefficient for each pixel is N^2, where $N*N$ is the size of the mask, therefore it is expedient to choose a mask of a small size. If the size the test image is $M*M$, the overall complexity of the algorithm is $O(M^2N^2)$.

2D cross-correlation is a time-consuming computational process, so it has to be optimized.

Preprocessing may be used to improve the computation process for mean of the image pixels covered by the mask. Instead of recalculating the sum of the pixels and dividing it by its quantity each step, it is advantageous to create a matrix that stores the sums of elements in the direction from left to right and from top to bottom of a given cell calculated beforehand. Then, to get the sum of the elements of a necessary submatrix we need to subtract the sums of smaller submatrices (those above and to the left) from the big submatrix, while taking care not to subtract the same elements twice. (Fig. 2).

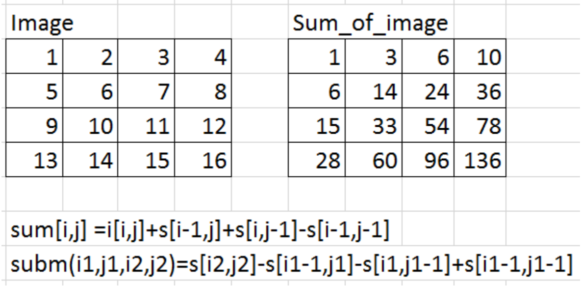

Image					Sum_of_image			
1	2	3	4		1	3	6	10
5	6	7	8		6	14	24	36
9	10	11	12		15	33	54	78
13	14	15	16		28	60	96	136

sum[i,j] =i[i,j]+s[i-1,j]+s[i,j-1]-s[i-1,j-1]

subm(i1,j1,i2,j2)=s[i2,j2]-s[i1-1,j1]-s[i1,j1-1]+s[i1-1,j1-1]

Fig. 2. The example of preprocessing

Thus, the complexity of getting sum of elements becomes $O(1)$ instead of $O(N^2)$, and the total algorithm complexity is $O(2M^2)$, or, from the rules of asymptotic complexity calculation, it is $O(M^2)$.

Limitations of the algorithm

Returning to the issue considered in the one-dimensional case, where the analyzed area goes beyond the boundaries of the image array, we can apply a similar solution to the two-dimensional case.

The first option is to consider missing pixels as zeros and do the same computation. The second option is to "wrap around" the image, using the formula $x(N + s)(N + t) = x(s)(t)$. In both cases, it is likely that the points of coincidence of image and the mask close to the image boundaries will not be found. Therefore, it is possible to ignore the computation of the boundary pixels at all.

Results

Approbation of the cross-correlation algorithm was performed using Microsoft Visual Studio 2013 and OpenCV 2.4.9. OpenCV (Open Source Computer Vision) library has *matchTemplate* function, containing methods for finding correspondences between the image and the template.

There are six matching methods that are available in OpenCV, and two of them use cross-correlation algorithm. They are called *CV_TM_CCORR* and *CV_TM_CCORR_NORMED*.

CV_TM_CCORR method is similar to the un-normalized cross-correlation coefficient, but it performs only summation of pixels, without subtracting the mean of mask and image. This method is sensitive to data collision, and may produce incorrect results when the large mask is applied. The formula is shown below:

$$R(x,y)=\sum_{x'y'}[T(x',y')\cdot I(x+x',y+y')]$$

The problem of data collision is solved in the *CV_TM_CCORR_NORMED* method: [5]

$$R(x,y)=\frac{\sum_{x'y'}[T(x',y')\cdot I(x+x',y+y')]}{\sqrt{\sum_{x'y'}T(x',y')^2\cdot\sum_{x'y'}I(x+x',y+y')^2}}.$$

The *matchTemplate* function compares the pixels relative to upper left corner of the template and skips the problematic regions near the image boundaries.

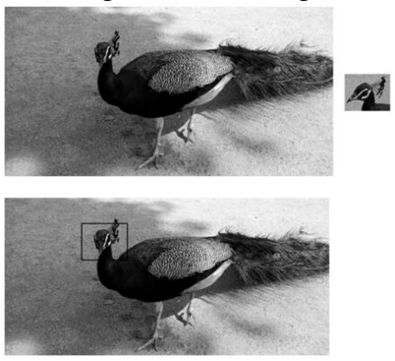

Fig. 3. The result of applying the algorithm

According to the results (Fig. 3), we observe that cross-correlation algorithm, despite its simplicity, is able to adapt to small distortions and interference, like changes in proportions and size and object's rotation.

References

1. Image registration. // Wikipedia [Electronic source]. – URL: http://en.wikipedia.org/wiki/Image_registration (30.09.2014).

2. Correlation and dependence. // Wikipedia [Electronic source]. – URL: http://en.wikipedia.org/wiki/Correlation_and_dependence (30.09.2014).

3. Pearson product-moment correlation coefficient. // Wikipedia [Electronic source]. – URL: https://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient (30.09.2014).

4. Paul Bourke. Cross Correlation. // paulbourke.net [Electronic source]. – URL: http://paulbourke.net/miscellaneous/correlate/ (30.09.2014).

5. Template Matching. // OpenCV 2.4.9.0 documentation [Electronic source] – URL: http://docs.opencv.org/doc/tutorials/imgproc/histograms/template_matching/template_matching.html (30.09.2014).