A TRANSFORM METHOD FOR FAST GENERALIZED IMAGE REGISTRATION

Yi Wan *Wah Chiu*

National Center for Macromolecular Imaging
Verna and Marrs McLean Department of Biochemistry and Molecular Biology
Baylor College of Medicine
One Baylor Plaza
Houston, TX 77030

ABSTRACT

We present an image transform and show that it is useful for fast generalized image registration. In particular, for an image of the size N pixels, this transform takes $O(N)$ additions/subtractions and $O(N_1)$ multiplications where N_1 is the bigger of the image height and width (and equals $N^{1/2}$ for a square image) – much faster than the FFT. Unlike FFT, whose speed depends on how the image size is factored, this transform has no such limit. Furthermore, for any invertible linear transform, including rotation, reflection, scaling, plus any translation, this transform can be used to match an image with its thus transformed image without any interpolation and with the above mentioned complexity. This transform can be applied naturally to signals of any dimension. Experiments show the advantages of this method.

1. INTRODUCTION

Many image registration problems [1, 2, 3] can be formulated as the following. Given an image x and another image y obtained from x through some transform T, find such T. The transform T can be translation, rotation, a combination of the two, etc. The simplest method in pattern matching is to take the (normalized) inner product, which is the same as matched filter if we view x as a filter. To apply this approach to match x and y, one usually applies all possible transforms T to x and then match the result to y. This is very time-consuming if the search space of T is large. In the case of translation only, FFT can be used to reduce computation complexity from $O(N^2)$ to $O(N \log_2 N)$ for some N values. However, in the case of rotations, it becomes much more complicated. Although spherical basis functions can be used to facilitate the use of FFT, interpolation often introduces error. Furthermore, the possible reflection or scaling in a single direction make this approach even more difficult.

In this paper we present another approach which is based on an image transform and overcomes the difficulties mentioned above. We first consider the continuous signals and view an image as a function on the plane $\mathbb{R}^2$ and study the following generalized transform,

$$Tx(t) = x(A(t-a)) \tag{1}$$

where A is an invertible 2×2 real matrix and t and a are vectors of size 2×1.

This class of transforms includes translations, rotations, reflections, and scaling in any one or two directions. We call image matching on this set of transforms generalized, as compared to the same problem on more restricted set of transforms. In the rest of the paper we present a method to efficiently find such transform and discuss its applications.

2. THE TRANSFORM METHOD

Given any x and y such that $y(t) = x(A(t-a))$, Let H be the set of functions defined as

$$H = \left\{ h : \mathbb{R} \to \mathbb{R}, \ \left| \int h(x(t))t dt \right| < \infty \right\}.$$

Note that in the above definition, the integration inside $|\cdot|$ is a 2×1 vector.

For any function $h \in H$, Define

$$u(h,x) = \int h(x(t))t dt \tag{2}$$

and

$$v(h,x) = \int h(x(t)) dt. \tag{3}$$

Then it can be derived that

$$u(h,y) = \frac{1}{|A|}\left(A^{-1}u(h,x) + v(h,x)a\right). \tag{4}$$

This work was supported by grants from NIHP41RR02250 and Agouron Institute and a training fellowship from the W. M.Keck Foundation to the Gulf Coast Consortia through the Keck Center for Computational and Structural Biology.

Note that in (4) we have a linear equation with 6 variables, 4 from the matrix $A^{-1}/|A|$ and 2 from the vector $a/|A|$. By changing h, we get other such linear equations, then a simple least square estimate can be performed to get

$$B = \frac{1}{|A|}A^{-1} \tag{5}$$

and

$$b = \frac{1}{|A|}a \tag{6}$$

and the final estimates for A and a are obtained as

$$\hat{A} = |B|^{1/3}B^{-1} \tag{7}$$

and

$$\hat{a} = |B|^{-1/3}b. \tag{8}$$

3. PRACTICAL CONSIDERATIONS

Although the set H is quite general, in practice there are at least three considerations that needs to be taken into account, namely, non-degeneracy, speed and accuracy.

The system (4) will be degenerate if $v = 0$, or x has zero mean. In this case (6) is no longer valid since b is indeterminable, and we cannot estimate a. Similarly, if h's are all linear functions, the system is also degenerate. To eliminate such situations, we can choose h to take the form $h(r) = |f(r)|$ where f is some function and $r \in \mathbb{R}$.

In computer processing speed is always an important issue. To increase speed, effort is often made to use simple operations such as addition/subtraction and logical and/or as much as possible. So we can choose h as $h(r) = |r - c|$ where c is any constant.

Furthermore, suppose the background intensity value is 0, then we also need to make h continuous at 0 so that $h(x(t))t$ will be integrable on $\mathbb{R}^2$. And taking into consideration of the various errors such as the image sampling error, it's good to make h continuous. With all these considerations, and without loss of generality assuming that x and y are nonnegative, we can choose the following set

$$H = \left\{ h : h(r) = \begin{cases} r, & r < c \\ |2c - r|, & r \geq c \end{cases}, \; r \in \mathbb{R}^+ \right\} \tag{9}$$

where $\mathbb{R}^+$ is the set of non-negative real numbers.

4. EXPERIMENTS

We present the experimental results to show the advantages of the algorithm developed in this paper. In the experiments, each pixel size is taken to be 1×1 unit. The integrations in (2) and (3) are taken as

$$\begin{aligned} u(h,x) &= \sum_{t_1,t_2} h(x(t_1,t_2)) \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \\ &= \begin{bmatrix} \sum_{t_1} t_1 \sum_{t_2} h(x(t_1,t_2)) \\ \sum_{t_2} t_2 \sum_{t_1} h(x(t_1,t_2)) \end{bmatrix} \end{aligned} \tag{10}$$

and

$$v(h,x) = \sum_{t_1,t_2} h(x(t_1,t_2)). \tag{11}$$

By using the h functions chosen from the set H defined in (9), we see that the computation of $u(h,x)$ in (10) needs $O(N)$ additions/subtractions and $O(N^{1/2})$ multiplications for a square image of a total of N pixels.

First we note that if the transform is translation only, i.e., the matrix A is identity, then there are only two parameters and we can easily compute the estimate $\hat{a}$ as

$$\hat{a} = \frac{u(h,y) - u(h,x)}{v(h,x)}. \tag{12}$$

This simple case is simulated and we always get $\hat{a} = a$. An example is presented in Figure 1, where the upper left 256×256 portion of an 1000×1000 image is translated by $[700, 500]^T$.

Next we present a more general case where A is a rotation matrix multiplied by a scaling factor. A 256×256 test image is first embedded in the center of an arbitrary 1000×1000 background image to get x. Then for every 10 degrees from 0 to 180, it is first scaled by a factor s randomly chosen in the range $[1/1.2,\ 1.2]$, rotated counterclockwise by that degree through nearest-neighbor interpolation, then translated by the vector a whose two integer numbers are randomly chosen from the range $[-200,\ 200]$. The rightward and downward directions are the positive directions of the coordinate system.

For each choice of h (or c), we get two equations. There are a total of 6 (correlated) parameters, so we can choose to apply (4) at least 3 times. We choose 3 in the experiment with the 3 choices of c value equally spaced in the pixel value range. When running on a Pentium 4/2.4GHz processor as a C++ program, it takes about 7 seconds to complete the 19 matchings. The parameters used to generate y and the estimated parameters are tabulated in Table 1. The images x and y for the rotation angle $\theta = 40^\circ$ are presented in Figure 2.

From Table 1 we see that the estimation is fairly accurate. For larger range of scaling, we find that the estimation errors also start to increase. This is possibly because of the

larger interpolation error during the scaling process in the generation of y.

5. DISCUSSION

The transform we present in this paper proves to be effective and efficient for a wide range of transforms in image matching and registration. We next discuss some extensions and limitations of this transform.

We first see that this transform is very effective for registration in an image database. All we need to do is to attach a vector of size $2n$ $(n \geq 3)$ to each image, then given any image we compute its u and v values and do a 6-parameter least square fit for each image in the database. Any fit with near 0 square error is a match. Also, the function x doesn't have to be single valued as in the gray scale image case. If x is a vector as in the color image situation, the transform can be readily applied to each element of x and the computation complexity remains the same. For example, for a gray scale image, we need to choose at least 3 different h functions to estimate A and a. For an RGB color image of the same size, we only need one h function to be applied to the 3 color bands of the image. The total computation cost is not changed. In addition, this transform naturally extends to higher dimensions. For a k-dimensional object matching problem, there are $k^2 + k$ transform parameters to be estimated, k^2 for A and k for a. The transform (3) and (4) can be computed only $k + 1$ times. In terms of the number of multiplications, for a general k dimensional signal with length $n_1, n_2, \cdots, n_k$ in each dimension, the total number of multiplications is $(k + 1)(n_1 + n_2 + \cdots + n_k)$. This is very efficient compared to FFT.

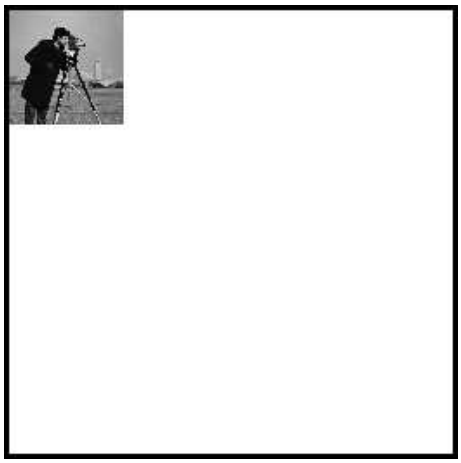
A major drawback of this transform is that it cannot be directly applied to use a template to pick out an object from a scene where other objects not from the template are present. One simple way to get around this problem is to first segment the scene and then do a registration problem on the segments. We point out that if in the scene all the objects are variously transformed objects from the same template, the method in this paper still can be applied directly. The only difference is that we now have $6m$ parameters to estimate where m is the number of objects, and hence the number iterations is also multiplied by m.

6. REFERENCES

[1] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, no. 4, pp. 325–376, 1992.

[2] B. S. Reddy and B. N. Chatterji, "An fft-based technique for translation, rotation and scale-invariant image registration," *PAMI*, vol. 5, no. 8, pp. 1266–1270, 1996.

[3] P. Thevenaz, U. E. Ruttimann, and M. Unser, "A pyramid approach to subpixel registration based on intensity," *IP*, vol. 7, no. 1, pp. 27–41, 1998.



(a) Image x

(b) Image y

Fig. 1. Translation estimation. Image y is x translated right and down by 700 and 500 pixels respectively. The estimates from (12) are also 700 and 500.

θ	s	a_1	a_2	$\hat{\theta}$	$\hat{s}$	$\hat{a}_1$	$\hat{a}_2$
0	0.89	-95	116	0	0.88	-95	115
10	0.92	-19	37	10	0.92	-19	37
20	1.00	59	34	20	1.00	59	34
30	0.94	-31	147	27	0.96	-32	150
40	1.17	-115	-171	36	1.18	-118	-176
50	0.86	-146	-134	50	0.86	-146	-133
60	0.91	110	11	60	0.91	110	11
70	1.19	-142	81	70	1.19	-142	81
80	1.10	79	31	82	1.10	79	31
90	1.18	-49	38	90	1.18	-48	37
100	1.11	-42	96	100	1.12	-43	97
110	1.14	-21	-123	108	1.14	-21	-122
120	1.18	-133	88	122	1.18	-133	87
130	0.85	-55	-142	133	0.85	-55	-145
140	1.10	77	-57	140	1.10	77	-57
150	0.85	38	-51	150	0.85	38	-51
160	0.85	-82	-188	160	0.85	-82	-188
170	0.99	-196	-130	169	0.98	-195	-129
180	0.94	33	101	180	0.93	33	98

Table 1. Image registration simulation result. θ is the rotation angle in degrees, s is the scaling factor, a_1 and a_2 are the translation values in the horizontal and vertical directions respectively. The right half of the table lists the estimated values for those parameters from the algorithm in this paper.

(a) Image x

(b) Image y

Fig. 2. Example of a transformed image. y is obtained by shrinking x to $\frac{1}{1.17}$ the original size, followed by rotating it counter-clockwise by 40 degrees, and then translated to the left and up by 115 and 171 pixels respectively. The matching algorithm developed in this paper estimates those values as $\frac{1}{1.18}$, 36 degrees, 118 and 166 respectively.