Fast Image Registration Using a Multi-Pass Image Interpolation Approach

An Hung Nguyen, Mark R. Pickering and Matthew A. Garratt

Abstract – Image registration is a fundamental technique in image processing. It is used to spatially align two or more images that have been captured at different times, from different sensors, or from different viewpoints. There have been many algorithms proposed for this task. The most common of these being the well-known Lucas-Kanade and Horn-Schunk approaches. However, the main limitation of these approaches is the computational complexity required to implement the large number of iterations necessary for successful alignment of the images. In this paper, we present an alternative approach for image registration using a modified version of the Image Interpolation Algorithm (I2A). Our proposed approach requires far fewer iterations to successfully register two images than the standard Lucas-Kanade approach. This means that our approach is much more suitable for pipelined hardware implementations that are required in real-time FPGA-based registration applications.

Index Terms – Global motion estimation, image interpolation algorithm, optical flow, image registration.

I. INTRODUCTION

The estimation of motion or optical flow between images has attracted much research interest over the past three decades. The main approaches for computing optical flow can be classified into four forms: gradient based methods, region based matching, energy-based methods and phase based methods [7]. The gradient based techniques emerge as the most efficient with low computational cost suitable for working in real-time processing. Take for example, the methods of Horn-Schunck [4] and Lucas-Kanade [5]. The first technique, which assumes smoothness in the flow over the whole image, takes the first derivative of the images to produce a flow velocity vector field. The derivative is approximated by the local gradient at each pixel and is calculated by taking the difference between adjacent pixels in both space and time. The velocity estimates are obtained by iteratively solving a set of simultaneous equations. More iterations give more accurate results at the cost of longer computation time and latency. The second technique also uses the first derivative, but with the assumption of constant flow in a local neighbourhood of the pixel under consideration. The velocities for each pixel are obtained by finding the least-square error value within the region. This technique is more accurate in regions containing different velocities. However, it does not fill in the centre of smooth regions and produces much lower flow densities as a result. This algorithm also provides more confident parameters for pixel velocities, compared to Horn-Schunck’s approach.

As an alternative to these iterative approaches, our group has previously proposed using the image interpolation algorithm (I2A) for implementing a real-time optical flow sensor using an FPGA [2]. The application for this work was the development of a small and lightweight sensor suitable for mounting on a small unmanned flight vehicle (a model helicopter in our case). However, the I2A in its original form is not an iterative procedure. It is designed to produce a once-off estimate of the motion between frames. This approach is only suitable for the case of small translations and rotations between frames of a video captured by the sensor. To partially solve this problem, an extension of the I2A approach called the iterative image interpolation algorithm (I3A) was developed [3]. This method estimates the size of the next image shift \( S_{k+1} \) based on previous image shift \( S_k \) and the measured optical flow \( Q \) using the following equation:

\[
S_{k+1} = γ (S_k + Q) + (1 - γ)S_k
\]

where \( γ \) is a parameter between 0 and 1.0.

The expected shift used in this computation will help overcome the limited range of motion that can be measured by the I2A algorithm. However, this method only deals with translations and can be unstable for unexpected changes in the trajectory of the unmanned vehicle.

In this paper, we propose a method to register two images by applying a multi-pass version of the I2A algorithm (MP-I2A) using a small number of iterations. Our experimental results show that this algorithm provides much greater accuracy than the original single iteration I2A approach and requires far fewer iterations than the standard Lucas-Kanade approach for the same registration accuracy.

The remainder of the paper is organized as follows. In Section II, we describe the original I2A approach. In Section III, we describe our patch-based approach for estimating affine motion using I2A. In Section IV, we describe our multi-pass I2A approach and present results for our experiments to choose the best parameters for this approach. In Section V, we compare our approach with the standard Lucas-Kanade
algorithm and in Section VI we draw conclusions from the results of our evaluation and outline areas for future work.

II. THE IMAGE INTERPOLATION ALGORITHM (I2A)

The main idea behind the I2A is that it is possible to interpolate an image at time \( t \) by using the image at time \( t_0 \) and six reference images formed by rotation and translation of the image at time \( t_0 \) [1]. The algorithm then requires the solution of a three-equation system to compute translational and rotational parameters of the motion between the two images. However, for small rotations between images, we will use a simpler version of the algorithm. This version is presented as follows:

Consider two snapshots or video frames, \( f_0(x,y), f(x,y) \) of a moving plane which is captured at the moments \( t_0 \) and \( t \). From \( f_0 \), four new images, \( f_1, f_2, f_3, f_4 \) are created by translating \( f_0 \) by amounts \( \Delta x_{ref} \) and \( \Delta y_{ref} \) along the \( x \) and \( y \) axes, respectively. For instance,

\[
f_1(x,y) = f_0(x + \Delta x_{ref}, y)
\]

\[
f_2(x,y) = f_0(x - \Delta x_{ref}, y)
\]

\[
f_3(x,y) = f_0(x, y + \Delta y_{ref})
\]

\[
f_4(x,y) = f_0(x, y - \Delta y_{ref})
\]

When the image deforms continuously and linearly from \( f_0 \) to \( f \), \( f \) can be interpolated from \( f_0 \) and the four reference images \( f_1, f_2, f_3, f_4 \) using the following formula:

\[
\hat{f} = f_0 + 0.5 \left( \frac{\Delta x}{\Delta x_{ref}} \right) (f_2 - f_1) + 0.5 \left( \frac{\Delta y}{\Delta y_{ref}} \right) (f_4 - f_3)
\] (1)

To compute the image’s motion between time \( t_0 \) and \( t \), the values of \( \Delta x \) and \( \Delta y \) are calculated so that the error between \( \hat{f} \) and \( f \) is minimized. An estimation of the plane motion is performed within a patch centered on the origin \( x = 0, y = 0 \) of a Cartesian system of coordinates. The size and shape of this patch are specified by a window function, for example:

\[
\psi(x,y) = \exp \left\{ -\frac{2.771}{p^2} (x^2 + y^2) \right\}
\] (2)

Where: \( p \) is the half-width of the Gaussian function (in pixels). The mean square error over the patch:

\[
E = \iint \psi[f - \hat{f}]^2 \, dx \, dy
\] (3)

is then minimized. This is equivalent to minimizing:

\[
E = \iint \psi \left[ f - \left( f_0 + 0.5 \left( \frac{\Delta x}{\Delta x_{ref}} \right) (f_2 - f_1) + 0.5 \left( \frac{\Delta y}{\Delta y_{ref}} \right) (f_4 - f_3) \right) \right]^2 \, dx \, dy
\] (4)

By setting the partial derivatives of \( E \) with respect to \( \Delta x \) and \( \Delta y \) to zero, we obtain the following two equations:

\[
\frac{\Delta x}{\Delta x_{ref}} \iint \psi (f_2 - f_1) \, dx \, dy + \frac{\Delta y}{\Delta y_{ref}} \iint \psi (f_4 - f_3) \, dx \, dy = 0
\]

\[
\iint \psi (f_2 - f_1)(f_2 - f_1) \, dx \, dy + \iint \psi (f_4 - f_3)(f_4 - f_3) \, dx \, dy = 0
\]

These two equations are linear with two unknowns, \( \Delta x \) and \( \Delta y \). Solving the system gives the translation between the two images.

Our approach is to estimate translational parameters for pairs of small non-overlapping patches in the two images by solving the above two-equation system. Although each pair of parameters only stores translational motion, the combination of rotational information can be estimated from the local translation of smaller patches in the image. This approach is illustrated in Fig. 1 which shows the local patch translation vectors for a global rotation between two images.

III. AFFINE MOTION USING THE I2A

Our proposed algorithm for estimating the global affine motion between two images using the I2A algorithm can be summarized as follows:

1) Pre-filter the input images to satisfy smoothness constraints.

2) Divide the images into patches (in this paper, the patch size is 64x64 pixels).
Fig. 2. The image used in our experiments is typical of that captured by a downward facing camera attached to our unmanned model helicopter.

3) Apply the I2A to each pair of patches in the two images to calculate translational parameters (optical flow) between them.

4) For the calculated optical flow vectors of the entire image, select \( n \) \((n \geq 3)\) sets of translational motion parameters to compute the global affine parameters of the transformation.

Steps 3 and 4 in the above procedure are explained in more detail as follows:

Assume \( f_0 \) and \( f \) are two patches and \( f_1, f_2, f_3, f_4 \) are reference images as presented in section II. According to (5) and (6), and replacing the double integrals by summation, we have the following formulas:

\[
A_1 = \sum (f_2 - f_1)^2 / \Delta x_{\text{ref}}
\]

\[
B_1 = \sum (f_2 - f_1)(f_3 - f_1) / \Delta y_{\text{ref}}
\]

\[
C_1 = 2 \cdot \delta (f - f_0)(f_2 - f_1)
\]

\[
A_2 = \sum (f_2 - f_1)(f_4 - f_3) / \Delta x_{\text{ref}}
\]

\[
B_2 = \sum (f_4 - f_3)^2 / \Delta y_{\text{ref}}
\]

\[
C_2 = 2 \cdot \delta (f - f_0)(f_4 - f_3)
\]

Therefore, the following two-equation system is established.

\[
\begin{bmatrix} A_1 & B_1 \\ A_2 & B_2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}
\]

(13)

Solving the above system, we have two parameters, \( \Delta x \) and \( \Delta y \) which describe the translational motion between the two patches.

\[
\Delta x = (C_1 B_2 - C_2 B_1) / (A_1 B_2 - A_2 B_1)
\]

\[
\Delta y = (A_1 C_2 - A_2 C_1) / (A_1 B_2 - A_2 B_1)
\]

(14)

(15)

After the translational motion for each patch is calculated, we choose \( n \) points (three points at least) to regress the global affine transformation. We perform the following procedures:

1) Select \( n \) starting points, \((x_1, y_1), (x_2, y_2), ...\), \((x_{n-1}, y_{n-1}), (x_n, y_n)\) located at the centre of the patches.

2) Compute \( n \) stopping points \((x_1', y_1'), (x_2', y_2'), ...\), \((x_{n-1}', y_{n-1}'), (x_n', y_n')\) corresponding to the starting points plus their translational motion.

\[
x_1' = x_1 + \Delta x_1; \quad x_2 = x_2 + \Delta x_2
\]

\[
x_{n-1}' = x_{n-1} + \Delta x_{n-1}; \quad x_n = x_n + \Delta x_n
\]

\[
y_1' = y_1 + \Delta y_1; \quad y_2 = y_2 + \Delta y_2
\]

\[
y_{n-1}' = y_{n-1} + \Delta y_{n-1}; \quad y_n = y_n + \Delta y_n
\]

3) Solve the three-equation system:

\[
Xa = Y \Rightarrow a = X^{-1}Y
\]

\[
X = \begin{bmatrix} x_1 \ y_1 \ 1 \\ x_2 \ y_2 \ 1 \\ ... \ ... \ 1 \\ x_{n-1} \ y_{n-1} \ 1 \\ x_n \ y_n \ 1 \end{bmatrix}, \quad Y = \begin{bmatrix} x_1' \ y_1' \ 1 \\ x_2' \ y_2' \ 1 \\ ... \ ... \ 1 \\ x_{n-1}' \ y_{n-1}' \ 1 \\ x_n' \ y_n' \ 1 \end{bmatrix}
\]

\[
a = \begin{bmatrix} a_{11} \ a_{12} \ a_{13} \\ a_{21} \ a_{22} \ a_{23} \\ a_{31} \ a_{32} \ a_{33} \end{bmatrix}
\]

(16)

4) The transpose of the matrix \( a \) describes the global affine transformation between the two images.

IV. MULTI-PASS I2A (MP-I2A)

In our proposed multi-pass I2A approach we extend the standard I2A approach to allow more than one iteration for each pair of image patches. In a similar manner to the gradient-descent approaches, we update the image \( f \) after each iteration using the estimated affine parameters. This updated version is then used in the following iteration. In addition we allow a different low-pass filtering and \( \Delta x_{\text{ref}} \) and \( \Delta y_{\text{ref}} \) for the first three iterations.

The reference shifts used in our proposed algorithm were 16 pixels in the first iteration, 8 pixels in the second iteration and 4 pixels in the remaining iterations. The low-pass filter used for each of the first three iterations was a simple 2-dimensional averaging (or boxcar) filter with varying size. The optimal size of the averaging filter for each iteration was determined experimentally. We performed these experiments for two specific types of motion between patches: translation only and rotation only. The experiments were performed using the image shown in Figure 2. This image was chosen as the test image because the ultimate application of our proposed approach is for estimating the position of an unmanned model helicopter using a downward facing camera. This image is typical of the images captured using this camera.

The experiments were conducted by applying a known translation or rotation to the image in Fig. 2 to create the image \( f \) and then registering this image with the original image \( f_0 \). As part of this process we estimate the amount of translation or rotation that has been applied to the image.

The range of translation used in the experiments was from -30 pixels to +30 pixels for both horizontal and vertical directions. In order to simulate the worst case situation, the translation was applied in both directions simultaneously.

Fig. 3(a) shows the average mean-squared-error (MSE) between the updated version of \( f \) after one iteration and the original image \( f_0 \) for each amount of translation. Each curve on
this graph corresponds to a different size of the averaging filter applied to both \( f \) and \( f_0 \) before the translations were estimated. Fig. 3(a) shows that the best performance (the lowest MSE) is obtained when an averaging filter with a size of 39x39 pixels is used to pre-filter the images. Fig. 3(b) shows the same curves for the case when only rotation is applied to the image. In this case the rotation applied ranged from -0.3 to +0.3 radians (or approximately -17 to +17 degrees). The curves in Fig. 3(b) support the conclusion that a filter size of 39x39 pixels provides the best performance after one iteration.

Fig. 3(c) and Fig. 3(d) show the same curves after the second iteration with the size of the filter for the first iteration fixed at 39x39 pixels. These curves show that the best performance after two iterations was provided by a filter size
V. PERFORMANCE EVALUATION

To evaluate our proposed MP-I2A approach we compared its performance with that of the standard Lucas-Kanade gradient descent algorithm. We evaluated both registration algorithms using two performance criteria: a) the number of iterations required for successful registration and b) the percentage of successful cases after a given amount of iterations. These performance criteria were measured by applying a set of known translations and rotation to the image in Fig. 2 to create the image \( f \) and then registering this image with the original image \( f_0 \). For these experiments the transformed image was created using a combination of rotation and translation. The rotations applied were \(-10, -5, 0, 5\) and \(10\) degrees. For each of these rotations the image was translated by \(-10, -5, 0, 5\) and \(10\) pixels horizontally and for each horizontal translation the image was translated by \(-10, -5, 0, 5\) and \(10\) pixels vertically. Hence, the overall set of transforms applied to the original image consisted of 125 different combinations of rotation and translation with the maximum transform applied being \(10\) degrees rotation plus \(10\) pixels translation in both the horizontal and vertical directions.

A. No. of iterations for successful registration

Fig. 4(a) shows the minimum, mean and maximum MSE between the updated version of \( f \) and the original image \( f_0 \) after a given number of iterations. The curves in Fig. 4(a) show that the minimum MSE is the same for both algorithms, however there is a marked difference between the two algorithms for the mean and maximum values of MSE. In particular notice that the maximum values of MSE for the MP-I2A approach are less than the mean value for the Lucas-Kanade algorithm. It can also be seen from these curves that the MP-I2A approach requires only \(6\) iterations to successfully register the test cases with the maximum initial displacement while the Lucas-Kanade algorithm requires \(16\) iterations.

B. The registration success rate

In order to compare the success rate of the two competing algorithms we defined successful registration to be when the value of the MSE between the registered and original images was less than \(20\). Fig. 4(b) shows the percentage of the set of \(125\) transforms applied that could be successfully registered after a certain number of iterations. It can be seen from these two curves that for the same number of iterations the success rate of the MP-I2A approach is much better than for the Lucas-Kanade algorithm. For example the success rate for the MP-I2A algorithm is \(95\)% after \(4\) iterations while the rate for the Lucas-Kanade algorithm is less than \(20\)%.

VI. CONCLUSIONS

In this paper we have presented a new multi-pass I2A approach for fast image registration. Our results show that, when compared to the standard gradient-descent approach, this new approach requires far less iterations to successfully register two images. Furthermore our multi-pass I2A allows images to be registered which have much greater initial displacements between them than the traditional single iteration I2A approach. Our new approach is therefore ideally suited to a pipeline implementation on hardware platforms such as field programmable gate arrays (FPGAs) where each iteration of the algorithm is required to be implemented using separate hardware resources.

Future work in this project will involve developing an implementation of our proposed approach using a Xilinx
Spartan FPGA and extending this algorithm to be robust to illumination changes between frames.

REFERENCES


