CS6320 Final Project: Image Sequence Stabilization with Optical Flow

Lingbing Jiang and Ting Liu

Abstract

Image sequence stabilization is widely used in many fields related to photography. In this project, we propose a three-step method to compensate for the influence of camera shake. First, we estimate the motion field with a state-of-the-art optical flow method; then the transformation parameter matrices between each two consecutive images are computed; finally, the whole image sequence is stabilized via piecewise smoothing. Experiment results indicate that our method is successful in stabilizing image sequences with/without moving foreground taken with significant camera shake.

1 Introduction

Image stabilization is a critical techniques used in a variety of fields from photography to astronomical telescopy [1]. It compensates the blurring or difference in captured images due to camera movement at the shutter moment. Generally, there are two types of scenarios where image stabilization is needed. One is single image stabilization, which removes blurring of a single image due to camera shake in still photography. The other one is, more accurately, image sequence stabilization or video stabilization, which aims at stabilizing or smoothing the motion of background in an image sequence. There are two kinds of image stabilization techniques, optical stabilization and digital stabilization. The difference can be easily told from their names. Optical techniques focus on the design of optical devices to reduce the influence of camera movement, while digital stabilization exploits the advantages of image processing and computer vision techniques and acts more as a post-processing step. In this project, we are concentrated on image sequence stabilization with a computer vision approach.

The fundamental idea of this project is to regard an image sequence as a series of image pairs and to use the motion fields reconstructed from such pairs to restore the image transformation parameters between two images in each pair. Then we can stabilize the whole image sequence by conducting serial transformations. Thus, three major problems are explored: the reconstruction of motion field from an image pair, the restoration of the transformation parameter matrix given motion field and the stabilization of an image sequence with transformation matrices. For motion

field reconstruction, we use a state-of-the-art algorithm for optical flow computation. Then we analyze the reconstructed motion field and solve for transformation parameters. Finally, a piecewise smoothing method is adopted for whole sequence stabilization.

2 Motion Field Reconstruction

Optical flow is a useful representation of object apparent motion caused by relative motion between an observer and the scene [2]. Numbers of algorithms are proposed for obtaining optical flow from image pairs. Let us consider the projection of scene as a function of two dimensional coordinates and time $I(x, y, t)$. By varying time t, we get an image sequence $\{I_t(x, y) : t = t_{\text{start}}, \ldots, t_{\text{end}}\}$. Since here we only consider the relative motion between a pair of images, we simply let the two images be I_1 and I_2. The very initial optical flow theory makes an assumption that the brightness difference across two images is zero, which is however not held in practice and modeled as an energy function (along with other additional terms) to be minimized. One of the most commonly used objective function is

$$E(\mathbf{u}, \mathbf{v}) = E_D(x, y, u_{x,y}, v_{x,y}) + \lambda E_S(x, y, u_{x,y}, v_{x,y}), \tag{1}$$

where the first term is called the data term and the second term is the smoothing term. They are respectively defined as

$$E_D(x, y, u_{x,y}, v_{x,y}) = \sum_{x,y} f_D\left(I_1(x, y) - I_2(x + u_{x,y}, y + v_{x,y})\right), \tag{2}$$

and

$$\begin{aligned} E_S(x, y, u_{x,y}, v_{x,y}) = \sum_{x,y} [& f_S(u_{x,y} - u_{x+1,y}) + f_S(u_{x,y} - u_{x,y+1}) \\ & + f_S(v_{x,y} - v_{x+1,y}) + f_S(v_{x,y} - v_{x,y+1})], \end{aligned} \tag{3}$$

where f_D and f_S are penalty functions for the data and the smoothing term respectively. In the implementation, the Charbonnier function $f(x) = \sqrt{x^2 + \epsilon^2}$ is used.

From equation 1, 2 and 3, we can see that the data term penalizes larger intensity difference between two corresponding pixels, and the smooth term prefers uniform motion within neighborhoods. By minimizing $E(\mathbf{u}, \mathbf{v})$ with respect to $\mathbf{u}$ and $\mathbf{v}$, we can obtain a motion field, i.e. a set of vectors $(\mathbf{u}, \mathbf{v})$ for each image pixel.

By using this objective function, we actually assume small motion between the two images, which is not always the case in practice. So the coarse-to-fine technique is often used. The idea is that by reducing the image resolution (building image pyramids) we can estimate larger motion in the same way as we estimate for small motion, because one pixel in a higher pyramid level corresponds to multiple pixels in a lower level.

The accuracy of optical flow estimation is crucial to the transformation restoration results, and false optical flow degrades the parameter estimation. The classical energy function formulation is straightforward, but it

does not always generate good results. There is an interesting paper [3] from CVPR 2010 that gives an thorough analysis of different models and methods used for optical flow estimation and explores the reasons behind different results. Moreover, an additional smoothing term based on median filtering is added to the objective function and proven successful. We try to follow their method and refine our optical flow estimation.

The major contribution the paper presents is modifying the objective function into

$$\begin{aligned} E(\mathbf{u},\mathbf{v}) =& E_D(x,y,u_{x,y},v_{x,y}) + \lambda E_S(x,y,u_{x,y},v_{x,y}) \\ &+ \lambda_2 E_C(\mathbf{u},\mathbf{v},\hat{\mathbf{u}},\hat{\mathbf{v}}) + \lambda_3 E_{NL}(\mathbf{u},\mathbf{v},\hat{\mathbf{u}},\hat{\mathbf{v}}), \end{aligned} \tag{4}$$

where E_C is the coupling term and E_{NL} is the non-local term. They are defined respectively as

$$E_C = \|\mathbf{u}-\hat{\mathbf{u}}\|^2 + \|\mathbf{v}-\hat{\mathbf{v}}\|^2, \tag{5}$$

and

$$E_{NL} = \sum_{x,y} \sum_{(x',y')\in N_{x,y}} \left(|\hat{u}_{x,y} - \hat{u}_{x',y'}| + |\hat{v}_{x,y} - \hat{v}_{x',y'}|\right). \tag{6}$$

Here $N_{x,y}$ is the set of all pixels in the neighborhood of (x,y). $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ denote an auxiliary flow field, which builds the connection to median filtering. [4] shows that

$$\hat{u}_{x,y}^{(k+1)} = \text{median}(N^{(k)} \cup D), \tag{7}$$

where $N^{(k)} = \{\hat{u}_{x',y'}^{(k)} : (x',y') \in N_{x,y}\}$, and $D = \{u_{x,y} \pm k\lambda_3/\lambda_2 : k = 0,1,\ldots,|N_{x,y}|/2\}$. Initially, we have

$$\hat{\mathbf{u}}^{(0)} = \mathbf{u}, \tag{8}$$

$$\hat{u}_{x,y}^{(1)} \approx \text{median}(N^{(0)} \cup \{u_{x,y}\}). \tag{9}$$

Similarly for $\hat{\mathbf{v}}$.

The coupling term acts as a connection in the process of optimization, which is done via alternately minimizing

$$\begin{aligned} E_O(\mathbf{u},\mathbf{v}) =& E_D(x,y,u_{x,y},v_{x,y}) + \lambda E_S(x,y,u_{x,y},v_{x,y}) \\ &+ \lambda_2 E_C(\mathbf{u},\mathbf{v},\hat{\mathbf{u}},\hat{\mathbf{v}}), \end{aligned} \tag{10}$$

and

$$E_M(\hat{\mathbf{u}},\hat{\mathbf{v}}) = \lambda_2 E_C(\mathbf{u},\mathbf{v},\hat{\mathbf{u}},\hat{\mathbf{v}}) + \lambda_3 E_{NL}(\mathbf{u},\mathbf{v},\hat{\mathbf{u}},\hat{\mathbf{v}}), \tag{11}$$

with respect to $(\mathbf{u},\mathbf{v})$ and $(\hat{\mathbf{u}},\hat{\mathbf{v}})$ respectively.

With $\mathbf{u}$ and $\mathbf{v}$ computed, we have the entire estimated motion field between two images.

3 Transformation Restoration

We learn from the computer vision class that camera movement leads to affine transformation between images. The transformation from one point on image 1 to one point on image 2 can be represented (in homogeneous coordinate) as

$$w \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \mathbf{P}_{1,2} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}, \tag{12}$$

where $\mathbf{P}_{1,2}$ is a transformation matrix in the form

$$\mathbf{P}_{1,2} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & 1 \end{bmatrix}. \tag{13}$$

By solving the scale factor w, we have

$$w = p_{31}x_1 + p_{32}y_1 + 1, \tag{14}$$

and thus

$$\begin{aligned} x_2 &= \frac{p_{11}x_1 + p_{12}y_1 + p_{13}}{p_{31}x_1 + p_{32}y_1 + 1} \\ y_2 &= \frac{p_{21}x_1 + p_{22}y_1 + p_{23}}{p_{31}x_1 + p_{32}y_1 + 1}, \end{aligned} \tag{15}$$

or

$$\begin{aligned} x_2 &= p_{11}x_1 + p_{12}y_1 + p_{13} - p_{31}x_2x_1 - p_{32}x_2y_1 \\ y_2 &= p_{21}x_1 + p_{22}y_1 + p_{23} - p_{31}y_2x_1 - p_{32}y_2y_1. \end{aligned} \tag{16}$$

Turning it into matrix representation and bringing in all the point pairs, we have

$$\begin{bmatrix} x_{11} & y_{11} & 1 & 0 & 0 & 0 & -x_{12}x_{11} & -x_{12}y_{11} \\ 0 & 0 & 0 & x_{11} & y_{11} & 1 & -y_{12}x_{11} & -y_{12}y_{11} \\ x_{21} & y_{21} & 1 & 0 & 0 & 0 & -x_{22}x_{21} & -x_{22}y_{21} \\ 0 & 0 & 0 & x_{21} & y_{21} & 1 & -y_{22}x_{21} & -y_{22}y_{21} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n1} & y_{n1} & 1 & 0 & 0 & 0 & -x_{n2}x_{n1} & -x_{n2}y_{n1} \\ 0 & 0 & 0 & x_{n1} & y_{n1} & 1 & -y_{n2}x_{n1} & -y_{n2}y_{n1} \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{31} \\ p_{32} \end{bmatrix} = \begin{bmatrix} x_{12} \\ y_{12} \\ x_{22} \\ y_{22} \\ \vdots \\ x_{n2} \\ y_{n2} \end{bmatrix}, \tag{17}$$

where $\mathbf{x}_{ij} = (x_{ij}, y_{ij})$ represents point i on image j, and n is the total number of point pairs. Abbreviate equation 17 as

$$\mathbf{Ap} = \mathbf{B}, \tag{18}$$

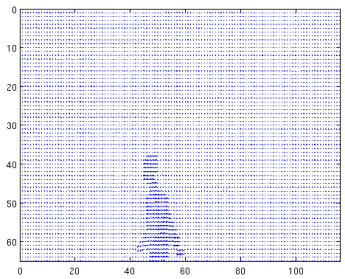
and then we can compute the transformation parameters simply as

$$\mathbf{p} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{B}. \tag{19}$$

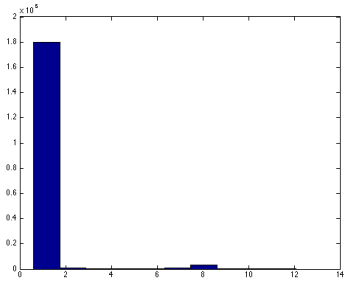
Equation 19 tells us we are able to restore the transformation parameter matrix given a set of corresponding points, which we obtain from the reconstructed motion field. Let us take a look at an example in figure 1. Figure 1(a) and 1(b) are two consecutive frames, between which the computed optical flow vector field is shown in figure 1(c). From the vector field, we can distinguish the foreground and the background easily. One clue that makes much sense here is that the foreground object has more motion than the background, which is actually also how we define foreground and background. Thus, we build the histogram of the motion vector magnitude as shown in figure 1(d). By implementing an adaptive threshold calculator, we can distinguish the foreground and the background pixels.

(a) Frame 150.

(b) Frame 151.



(c) Optical flow.



(d) Flow vector length histogram.

Figure 1: Reconstructed motion field example.

Since we would like to stabilize the image sequence with respect to the background, we use only the background flow vectors for transformation restoration. For a point in image 1 $\mathbf{x}_1 = (x_1, y_1)$ and its displacement vector $\mathbf{u} = (u, v)$, we can calculate its corresponding point in image 2 as

$$\mathbf{x}_2 = \mathbf{x}_1 + \mathbf{u}. \tag{20}$$

Combining this with equation 17 and then 19, we can fit the transformation parameter matrix between two images.

4 Image Sequence Stabilization

Next problem is to which framework we shall transform the image sequence with a series of transformation matrices. The simplest way is to globally smooth the sequence by transforming every image to the first image, which however can cause the accumulation of errors. If the optical flow estimation for one image pair is bad, all the images behind it will suffer from this incorrectness. Therefore, what we do is k-frame piecewise smoothing over total n frames:

1. Acquire the first frame I_1.
2. $m = 0$.
3. Acquire the next k frame $I_{km+2}, I_{km+3}, \ldots, I_{km+k+1}$. If frames not enough, acquire to the last frame. Denote the number of actually acquired frames as k'.
4. Compute $\mathbf{T}_m = \sqrt[k']{\prod_{i=2}^{k'+1} \mathbf{P}_{km+i-1,km+i}}$.
5. For $i = 2$ to $k' + 1$, $I^{\text{new}}_{km+i} = \mathbf{T}_m^{i-1}(\prod_{j=2}^{i} \mathbf{P}^{-1}_{j-1,j})(I_{km+i})$.[1]
6. $m = m + 1$.
7. If $km < n$, go to step 3; else terminate.

With this method, we can limit the errors in optical flow estimation and other numerical errors within a certain range. In practice, we choose $k = 32$, and we use bilinear interpolation during image transformation.

5 Results

The implementation is written in MATLAB. For optical flow estimation, we referred to the code provided by the paper author [5]. We implemented the code for the other parts. Also, we acquired the image sequences around Warnock Engineering Building by ourselves. We applied our method on four video segments.

The outputs are four corresponding stabilized video sequences. Due to the inefficiency of showing the image sequences in a text file, we submitted the four pairs of videos along with this report. Video 1 is an "easy" scenario in which there is moving foreground and the camera shake is not significant. Video 2 shows a scene without moving foreground but taken with significant camera shake. Video 3 has moving foreground and is taken with intense camera shake. Video 4 is taken with a moving camera with significant shake with no foreground presence. We recorded these four different scenarios in the hope of testing the performance of our method under different situations. According to the stabilized videos, our method is highly capable of achieving the stabilization task even for the sequences taken with significant camera shake. With video 4, which is experimental, we show that our method can also be used to smooth the image sequence taken with a moving camera.

[1] This means we apply a series of inverted $\mathbf{P}_{j-1,j}$ transformation on frame $km + i$ for $i - 1$ times and then apply $i - 1$ times $\mathbf{T}_m$ transformation.

6 Conclusion

In this project, we proposed a working pipeline for image sequence stabilization. Three steps are taken. First, the optical flows between each two consecutive images are estimated with a state-of-the-art method. Second, the transformation parameter matrix for each image pair is restored with the estimated motion field. Finally, a whole image sequence is stabilized with a piecewise smoothing scheme. Experiment results show that our method is capable of stabilizing image sequences with/without moving foreground taken even with significant camera shake.

References

[1] Wikipedia. *Image stabilization*, `http://en.wikipedia.org/wiki/Image_stabilization`.

[2] Wikipedia. *Optical flow*, `http://en.wikipedia.org/wiki/Optical_flow`.

[3] D. Sun, S. Roth and M.J. Black. *Secrets of optical flow estimation and their principles*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2432–2439.

[4] Y. Li and S. Osher. *A new median formula with applications to PDE based denoising*, Communications in Mathematics Sciences, 7(3):741-753, 2009.

[5] `http://www.cs.brown.edu/~dqsun/code/flow_code.zip`.