Journal of Signal and Information Processing, 2016, 7, 27-33
Published Online February 2016 in SciRes. http://www.scirp.org/journal/jsip
http://dx.doi.org/10.4236/jsip.2016.71004

Image Deformation by User-Defined Force Fields

Jian Wu

Department of Applied Mathematics, Jiangxi Agricultural University, Nanchang, China
Email: wudging@163.com

Received 10 January 2016; accepted 23 February 2016; published 26 February 2016

Abstract

We present a new method for image deformation. The warping technique provides smooth distortion with intuitive and easy manipulation. Driven by a restrained force field, the input image is deformed gradually and continuously. The method allows us to customize the force fields and the region of interest manually through some simple steps. Experimental results demonstrate the effectiveness and convenience of the approach.

Keywords

Image Deformation, Force Fields, Free-Form Deformation

1. Introduction

Image deformation has proven to be a powerful tool for image editing. There are many applications from animation, morphing, to image correction and recognition. In the last two decades, a considerable amount of research has been directed towards image deformation. The classical method for image deformation is obtained by linear combination of affine transformations; see [1] [2]. In the methods, the user needs a few handles and manipulates them to control the shape. The energy-based or variational-based methods are the powerful and popular way to image distortion. The approaches produce warps by minimizing a functional with some roughness criteria, such as [3] [4]. The techniques are always not easy to implement and mostly cost greater computation in the post-time. Free-form deformation methods (see [5]) and skeleton-based techniques (see [6]) distort the shapes by manipulating the space in which they are embedded. They are very efficient in computation and easier to be implemented, but they have not enough in providing convenient or intuitive tools for the user.

We want to propose here a new attempt for image deformation. The proposed method is intuitive, and easy to manipulate without tedious landmarks. We view image deformation as a result of a force field that is exerted on the input image. If the pixels of an image are regarded as small particles, under forces, the particles should be

How to cite this paper: Wu, J. (2016) Image Deformation by User-Defined Force Fields. *Journal of Signal and Information Processing*, **7**, 27-33. http://dx.doi.org/10.4236/jsip.2016.71004

moved and form new shapes. Here we want to simulate the distortion under external forces and extend it to image warping. We know that the Newton's Second Law of Motion (NSLM) links the force and displacement. Thus, through NSLM, we can infer the new position for each pixel by a given force field. We achieve the distortion in the form of a partial differential equation (PDE). PDE-based methods are widely applied in the image filtering, see [7]-[10]. The typical PDE-based approaches deform a given image with a specially-designed PDE, and obtain the desired result as the solution of this PDE with the input image as initial conditions. In the form of a PDE, one can model the images in a continuous domain, and control the evolving process more precisely to avoid over processing. In this paper, we present a new technique for image deformation. In the new method, we can customize the region of interest (ROI) to restrict distort in a desired region, while leaving the others without modification. Unlike the classical PDE-based filtering models, we attempt the useful tool to evolve an image on the coordinates or displacement, not on the value of intensity or the vector of color.

2. Deformation by User-Defined Force Fields

2.1. Image Deformation under Forces

Let $\boldsymbol{F}(x,y;t)$ be the force exerted on the pixel located at (x,y) at t time. According to NSLM, the net force on an object is equal to the rate of change of its linear momentum:

$$\boldsymbol{F} = \frac{\mathrm{d}(m\boldsymbol{v})}{\mathrm{d}t},$$

where m is the mass of the body, and $\boldsymbol{v}$ is instantaneous velocity. As the position is the prime consideration in image deformation, we ignore the differences between pixels and regard all the pixels have a constant-mass m. Notice that the velocity can be computed as the derivative of the displacement vector $\boldsymbol{S}$, then we get

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\boldsymbol{S}(x,y;t) = \frac{\boldsymbol{F}(x,y;t)}{m}, \tag{1}$$

where $\boldsymbol{S}(x,y;t)$ denotes the displacement of pixel (x,y) at t time. By the Equation (1), we can pass forces to image pixels and drive them as we desire. It allows us to apply it directly to image deformation by a specially-designed force field. Once the smooth force field is figured out, the image will be deformed continuously and naturally by the equation.

To solve the simple second-order PDE equation, we can certainly suppose that the initial position of each pixel is as that of the input image, and the velocity state is still. Therefore, we can calculate the new position for each pixel by

$$\begin{cases} \dfrac{\boldsymbol{S}_{n+2}(x,y) - 2\boldsymbol{S}_{n+1}(x,y) + \boldsymbol{S}_n(x,y)}{\Delta t^2} = \dfrac{\boldsymbol{F}_n(x,y)}{m}, \\ \boldsymbol{S}_0(x,y) = \boldsymbol{S}_1(x,y) = x + y\mathrm{i}. \end{cases} \tag{2}$$

However, the deformation driven by Equation (2) will leave some blanks on the image at each iteration, meanwhile; some coordinates perhaps contain more than one pixel. The two cases are inevitable. For the former, to remove these empty dots, there are many filters can be used. For example, we can use the classic average or median filtering, or the more effective methods, such as the inpainting method [11]. To lessen the disadvantage of this case, we can reduce time step and remove the blanks in time at each iteration. For the latter, if the deformation is ongoing, we reserve all the overlapped pixels for the next motion; else we must discard some of them, and only reserve one of them.

2.2. Design of Force Fields

Here we design two types of force fields to simulate the process of object distortion under external forces. One is for image stretching/squashing, and another for image twisting.

One can imagine how to stretch a thin piece of pizza. Suppose the pizza sticks on the table. We hold one position (here we call the begin pixel), pull and move it slowly to another (called the end pixel), and then we get it. Forces around the two pixels are stronger than that of far away the two pixels, and the orientation of the force can regard as the same one. For the sake of simplicity, let $\boldsymbol{F}(x,y;t)$ is constant with respect to time, *i.e.*,

$$\boldsymbol{F}(x, y; t) = \boldsymbol{F}(x, y), \text{ for any } t \geq 0,$$

and considering smooth and natural deformations, we define such a vector field $\boldsymbol{F}(x, y)$ as follows:

$$\left|\boldsymbol{F}(x, y)\right| = Strength_1 \cdot \exp\left[-\frac{(x - x_{beg})^2 + (y - y_{beg})^2}{2\sigma_1^2}\right] + Strength_2 \cdot \exp\left[-\frac{(x - x_{end})^2 + (y - y_{end})^2}{2\sigma_2^2}\right],$$

$$\angle \boldsymbol{F}(x, y) = \angle\left[S(x_{beg}, y_{beg}) - S(x_{end}, y_{end})\right],$$

where the pixels (x_{beg}, y_{beg}) and (x_{end}, y_{end}) are the begin pixel and the end pixel, respectively. The parameters σ_1 and σ_2 control, respectively, the scopes of deformation around the begin pixel and the end pixel, and the parameters $Strength_1$ and $Strength_2$ control the magnitude of force of the two pixels, respectively. In **Figure 1**, we show a force field which moving the pixel (130, 130) to the pixel (160, 160), and the corresponding deformed result. In this type of vector fields, we stop the distortion when the begin pixel is impelled to or

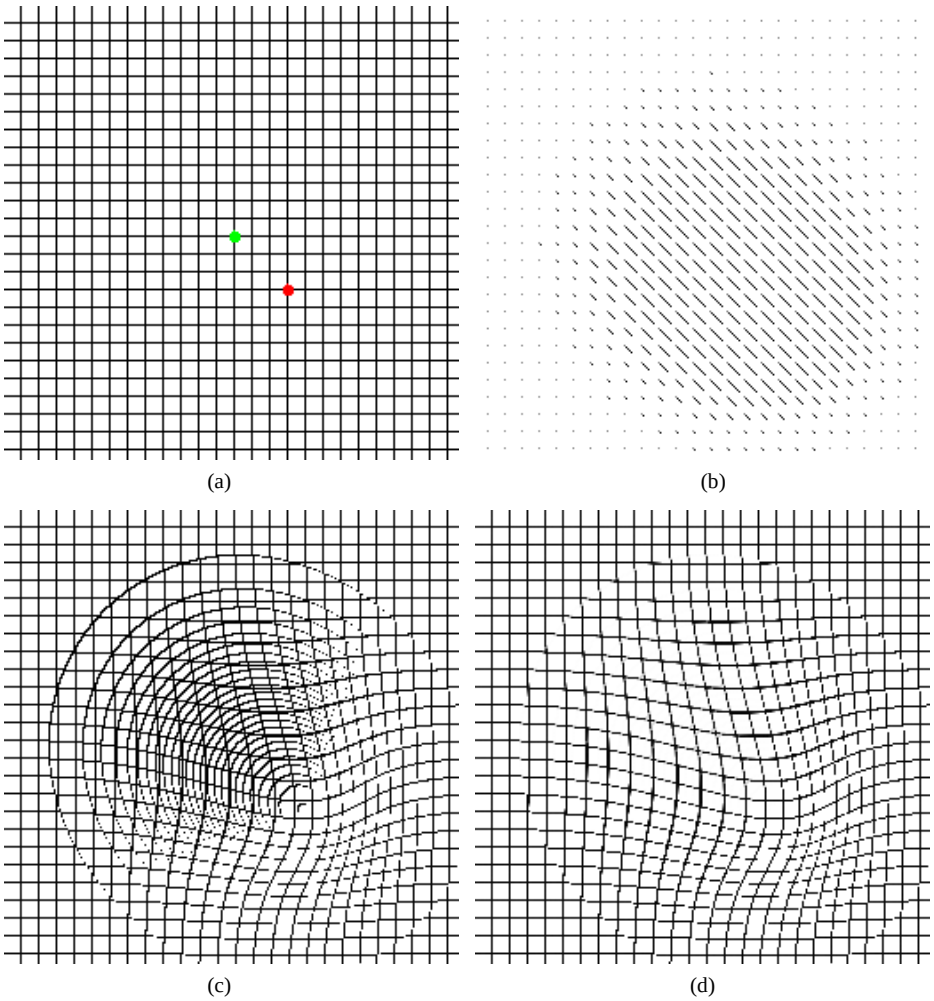


Figure 1. (a) Original image, 256 × 256 in pixels, with the begin pixel (green) and the end pixel (red); (b) force fields generated by Equation (1) with beg = (130, 130), end = (160, 160), $Strength_1 = Strength_2 = 20$, $\sigma_1 = \sigma_2 = 40$; (c) distorted image without filling blanks; (d) result with filling blanks.

over the location of the end pixel.

Similarly, we also employ the following force fields for image twisting:

$$\left|\boldsymbol{F}\left(x,y\right)\right| = Strength \cdot \sin\left[\frac{\sqrt{\left(x-x_{cen}\right)^2+\left(y-y_{cen}\right)^2}}{\sqrt{\left(x_{rim}-x_{cen}\right)^2+\left(y_{rim}-y_{cen}\right)^2}}\pi\right],$$

$$\boldsymbol{F}\left(x,y\right) \perp \left[\left(x+y\mathrm{i}\right)-\left(x_{cen}+y_{cen}\mathrm{i}\right)\right],$$

where $\left(x_{cen}, y_{cen}\right)$ and $\left(x_{rim}, y_{rim}\right)$ are, respectively, the center pixel and the rim pixel, which the two user-defined pixels are used to decide the central location and radius of twisting. In this type of force fields, we let the user decide when to stop the distortion.

Sometimes we want to restrict the deformation in a region of interest. We can select a region in the source image by drawing a simple closed curve as shown in **Figure 2(a)**. We zero the forces which are out of the region, and the forces within the region need to be modified to fulfill a smooth distort. In **Figure 2(a)**, we show a user-defined region, and **Figure 2(b)** show the smoothed one. In the smoothed mask, the values range from 0 to 1. The brighter a pixel is, the closer the mask value tends to 1; and the more black a pixel is, the smaller the value is. This smooth mask can be obtained by a morphological operator. First, we flood-fill the region with white color, and next apply an erosion operator to it. After that, we compare the result between the last step, and then set a different value for the eroded pixels. Repeat doing erosion, at last we can obtain the faded mask.

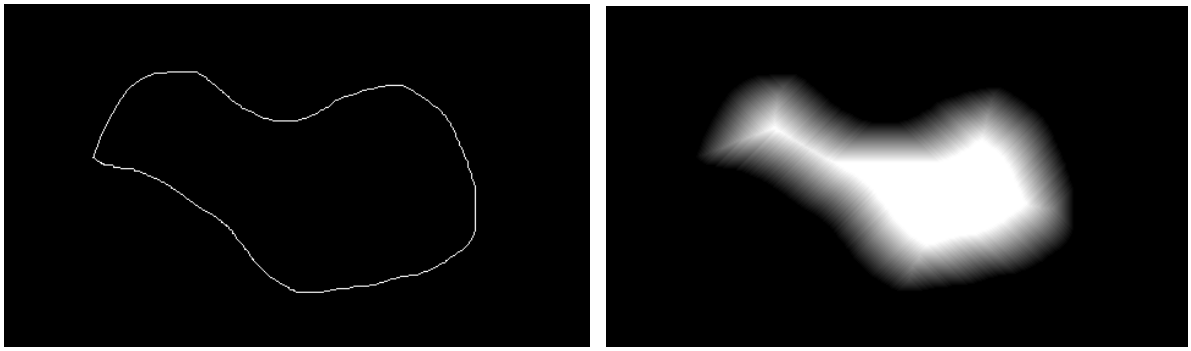
3. Experimental Results

In the experiments, we implement the new deformation with C++ code in the platform of Windows 7. We have developed an intuitive and convenient tool which is available at http://wudging.ys168.com/.

In the experiments, we set the begin and end pixels, or the center and rim pixel, and the deformation region of interest by mouse clicking or drawing. To reduce the amount of blanks when deforming, and to fulfill a better fill-in, we suggest that the time step should not be too big. In this paper, we always set $\Delta t^2 = 0.5$. As the blanks are unavoidable, it is necessary to employ a suitable filtering algorithm for the removal of them. Considering the run time and effectiveness, we choose the average filter with 5×5 local windows. In these experiments, some of the input images are manipulated in continuous processes, and we found that the time of each deformation is not more than 1 seconds.

Figure 3 shows the Lena image and her deformed hat, one can see that the surrounding objects, such the rods in the northwest and east of the hat, have no warps, because we have restricted the deformation into a free-drawn region that only contains the hat. **Figure 4** shows a deformed head. Additionally, by choosing a proper region of interest, we can deform the head without warping the eyebrows and the ears. **Figure 5** shows a wilder ocean wave by setting the center and rim pixels with the restricted region; from the result, we can see the deformation is smooth and hardly perceptible. From **Figure 6** and **Figure 7**, we can see more such examples.

In **Figure 8**, we show the proposed method can be applied to image correction. Usually, the texts of a book will be deformed when the book is open on the desk. The corrected result shows our method can be applied in such cases.

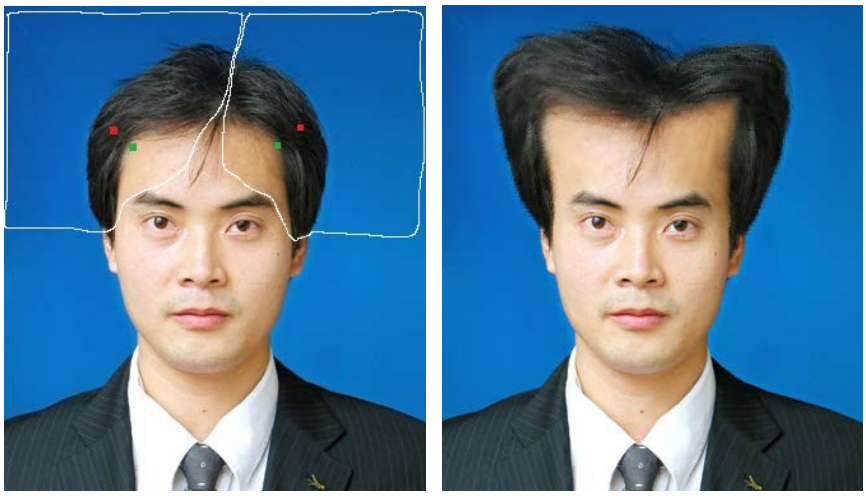


(a) (b)

Figure 2. Smooth mask for ROI. (a) An user-defined region; (b) the related mask.

(a) (b)

Figure 3. (a) Lena image, 512 × 512 in pixels; (b) her deformed hat.



(a) (b)

Figure 4. A deformed head. (a) Original image (331 × 300 in pixels) with the begin pixel (green) and the end pixel (red), and the regions of interest; (b) deformed image by two steps with the same parameters: $\text{Strength}_1 = \text{Strength}_2 = 80$, $\sigma_1 = \sigma_2 = 100$.

(a) (b)

Figure 5. A wilder ocean wave. (a) Original image (360 × 260 in pixels) with the landmarks, the center pixel (green), the rim pixel (red), and the region of interest; (b) deformed wave with the twisting force field: Strength = 100 and Iteration is 3.

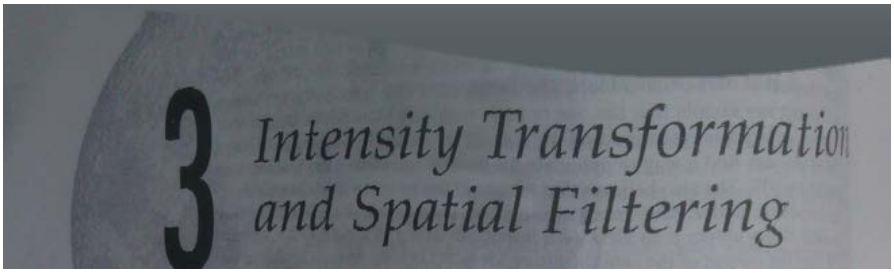
(a) (b)

Figure 6. A deformed alarm-clock. (a) Original (965 × 1016 in pixels); (b) deformed.

(a) (b)

Figure 7. A lovely flower. (a) Original image (529 × 458 in pixels); (b) deformed flower.

(a)



(b)

Figure 8. Removal of distortion. (a) A book is opening on the desk in the natural state. The picture is taken by a phone camera, (trimmed, 657 × 196 in pixels); (b) the corrected texts.

4. Conclusion

In this paper, we deform an image gradually and continuously using Newton's Second Law of Motion. The new method is in the interactive mode and very convenient. From the experiments, the deformed result of the proposed technique is very smooth and natural.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under 61561025 and 61562039.

References

[1] Schaefer, S., McPhail, T. and Warren, J. (2006) Image Deformation Using Moving Least Squares. *ACM Transactions on Graphics*, **25**, 533-540. http://dx.doi.org/10.1145/1141911.1141920

[2] Tang, Y.Y. and Suen, C.Y. (1993) Image Transformation Approach to Nonlinear Shape Restoration. *IEEE Transactions on Systems, Man, and Cybernetics*, **23**, 155-172. http://dx.doi.org/10.1109/21.214774

[3] Eitz, M., Sorkine, O. and Alexa, M. (2007) Sketch Based Image Deformation. *Proc. Vision, Modeling and Visualization* (*VMV*), 135-142.

[4] Karni, Z., Freedman, D. and Gotsman, C. (2009) Energy-Based Image Deformation. *Computer Graphics Forum*, **28**, 1257-1268. http://dx.doi.org/10.1111/j.1467-8659.2009.01503.x

[5] Yan, H.B., Hu, S.M., Martin, R.R. and Yang, Y.L. (2008) Shape Deformation Using a Skeleton to Drive Simplex Transformations. *IEEE Transactions on Visualization and Computer Graphics*, **14**, 693-706. http://dx.doi.org/10.1109/TVCG.2008.28

[6] Milliron, T., Jensen, R., Barzel, R. and Finkelstein, A. (2002) A Framework for Geometric Warps and Deformations. *ACM Transactions on Graphics*, **21**, 20-51. http://dx.doi.org/10.1145/504789.504791

[7] Perona, P. and Malik, J. (1990) Scale Space and Edge Detection Using Anisotropic Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**, 629-639. http://dx.doi.org/10.1109/34.56205

[8] Weickert, J. (1999) Coherence-Enhancing Diffusion Filtering. *International Journal of Computer Vision*, **31**, 111-127. http://dx.doi.org/10.1023/A:1008009714131

[9] You, Y.L. and Kaveh, M. (2000) Fourth-Order Partial Differential Equation for Noise Removal. *IEEE Transactions on Image Processing*, **9**, 1723-1730. http://dx.doi.org/10.1109/83.869184

[10] Tschumperle, D. and Deriche, R. (2005) Vector-Valued Image Regularization with PDEs: A Common Framework for Different Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**, 506-517.

[11] Wu, J. and Tang, C. (2011) An Efficient Decision-Based and Edge-Preserving Method for Salt-and-Pepper Noise Removal. *Pattern Recognition Letters*, **32**, 1974-1981. http://dx.doi.org/10.1016/j.patrec.2011.09.025