Control Methods for Fluid-Based Image Warping

Dalia Bonilla, Luiz Velho
IMPA-Institudo Nacional de Matemática Pura e Aplicada
www.impa.br/∼{dalia,lvelho}

Fig. 1. Result of our method: Frog animation from one image.

***Abstract*—Warping techniques can be complicated and difficult to use, but through the use of fluid dynamics the warping becomes simple and it is intuitively controlled by physical properties such as viscosity and forces. These properties are naturally associated with the image itself or with spatial control handles. The key idea is to think of the image domain as a two-dimensional incompressible and homogeneous fluid, and to use the Navier-Stokes equations to change it by applying forces to the image function. In this way, the process does not move the image values as in fluid simulations, but transforms the coordinates of a parametrization of the image through a vector field generated by the simulation equations effectively acting as a texture mapping. The contribution of this work is a new method for image warping based on fluid simulation and effective control mechanisms.** [1]

***Keywords*-warping, morphing, adjoint method.**

I. Introduction

Warping is a deformation process of objects in a image. The source image f is continuously deformed into another image g. The use of warping plays an important role in many applications, from the correction image distortions in medical data to the creation of special effects in the entertainment industry. Another application of warping is the technique known as *morphing*. Morphing use both warping and blending of two images j and h to create the change (or morph) between the image j into the image h.

The warping techniques, until now, are using transformations based on geometry to create deformations in image, with good results, but so far, few know techniques have exploited a deep study based on physics to create the deformations.

This paper proposes an alternative approach to image warping, spatial transformations based on Navier-Stokes equations, and the use of fluid dynamics in general, present a great potential in the above context because they are very powerful to derive warping transformations. In the present paper we develop a framework where such a technique is exploited and methods to control the warping are derived.

A. Related work

According to Gomes et al. [1], the warping methods, can be classified into parameter-based, feature-based, free-form based and hybrid techniques. The parameter-based methods encompasses all the warping techniques that are controlled by parameters, such as scale, twisting and bending. An early work using this technique was done by Alan Barr [2]. Feature-based methods cover a whole class of warping techniques, with a great variety of different geometric features. The warping is defined explicitly by mapping each feature in the source object to its correspondent in the target object. Image warping with scattered data interpolation techniques belongs to the class of the feature-based warping methods. A scattered data interpolation method was introduced by Arad and Reisfel [3]. Free-form-based warping techniques uses free-form curves (B-splines, Bézier, etc.) to define the warping transformations. An early example of these techniques was introduced by Smith [4].

Some important works in a long history in fluid simulation in computer graphics are the work of Foster and Metaxas, who used the full Navier-Stokes equations to model both water [5] and gases [6]. Stam [7] introducing the Stable Fluids algorithm, which combined semi-Lagrangian advection with an implicit viscosity solver. The work of Foster and Fedkiw [8] and Enright [9]. Here we mentioned two works on fluid control, Treuille et al [10] proposed the general framework for controlling smoke simulations using keyframing and nonlinear optimization. McNamara [10] greatly increased the speed of the optimization using the adjoint method. The adjoint method is used here to compute derivatives quickly and efficient. Some

[1]Extended abstract of doctoral thesis.

works are Giles and Pierce [11] discuss both the continuous and discrete approach. And Giering and Kaminski [12] give recipes for adjoint code construction.

The pioneer work using fluid dynamics in image processing was introduced by Bertalmio et al [13] with a method for digital inpainting. They think of the image intensity as stream function and the Laplacian of the image intensity plays the role of the vorticity of the fluid, which is transported into the region to be inpainted by a vector field. Our idea was presented in the work [14]. It is based also in the Navier-Stokes equations, but our technique, the fluid warping uses a vector field directly created by the velocity of the fluid by Navier-Stokes equations: we neither use the stream function nor vorticity.

In our work proposed fluid warping technique carries the coordinates of a parametrization of the image through of a vector field generated by the Navier-Stokes equations. There the warping is controlled by physical parameters such as viscosity and forces associated or not with the characteristics of the image itself or by other auxiliary images and applied to the dynamic simulation, here in this paper, also the warping is controlled by user-specified keyframes. A continuous quasi-Newton optimization solves for appropriate forces to be applied to the velocity field throughout the simulation. We use a method to efficient compute derivatives of a whole fluid simulation.

II. Fluid and Warping

In this part some mathematical concepts of fluid simulation are discussed. We model incompressible fluids, with a constant viscosity and a variable viscosity. This task require an efficient implementation, therefore we will employ the Stable Fluids algorithm and extend it to variable viscosity.

A. Basic Algorithm and Mathematical Formulation

We adopted and adapted the Stable Fluids algorithm for the implementation of the fluid simulation. This algorithm solves the Navier-Stokes equations, originally for the case of constant viscosity and incompressible fluid. This equations can be written in the following compact form:

$$\partial_t v = P(-v \cdot \nabla v + \mu \triangle v + f) \tag{1}$$

where v is the velocity of fluid, μ is the viscosity and f are the external forces. The operator $P(v) = w$ projects a vector field v onto its incompressible component w (divergence free, i.e, $\nabla \cdot w = 0$). For the case of variable viscosity we have the following Navier-Stokes equations in compact form:

$$\partial_t v = P(-v \cdot \nabla v + \nabla \mu(\boldsymbol{x})(\nabla v + \nabla v^\top) + \mu(\boldsymbol{x}) \triangle v + f) \tag{2}$$

where

$$\nabla v + \nabla v^\top = \begin{pmatrix} 2\partial_x v^1 & \partial_x v^2 + \partial_y v^1 \\ \partial_y v^1 + \partial_x v^2 & 2\partial_y v^2 \end{pmatrix}.$$

For the each time step $\triangle t$ the algorithm uses operator splitting and solves the equations in four steps. Starting form a velocity field v_0 of a previous time step, the algorithm decomposes the equations 1 sequentially:

$$v_0 \xrightarrow{\text{add force}} v_1 \xrightarrow{\text{advect}} v_2 \xrightarrow{\text{diffuse}} v_3 \xrightarrow{\text{project}} v_4$$

The first step is the addition of external forces f :

$$v_1 = v_0 + \triangle t f(\boldsymbol{x}, t).$$

The second step given by the advection equation

$$\partial_t v_2 = -(v_1 \cdot \nabla) v_2$$

which is solved with a semi-Lagrangian technique [15]

$$v_2(\boldsymbol{x}) = v_1(\boldsymbol{x} - \triangle t v_1(\boldsymbol{x}))$$

here it calculate a new velocity by interpolation.
The third step is diffusion

$$\partial_t v_3 = \mu \triangle v_3$$

that uses an implicit method, and gets the solution for the system

$$(I - \triangle t \mu \triangle) v_3 = v_2$$

by using the Jacobi method. The fourth step projects the velocity field onto the incompressible (divergence free) field. This step involves the solution of a Poisson equation $\triangle q = \nabla \cdot v_3$ and therefore we have $v_4 = v_3 - \nabla q$. The method for solving this stage are finite difference schemes and Jacobi.

We extend the formulation above to solve our equation 2 using the same stages, except that because we assume variable viscosity the diffusion stage will be different. More precisely, the original Stable Fluids solves the equation

$$\partial_t v_3 = \mu \triangle v_3$$

and we have to solve the equation

$$\partial_t v_3 = \nabla \mu(\boldsymbol{x})(\nabla v_3 + \nabla {w_3}^\top) + \mu(\boldsymbol{x}) \triangle v_3.$$

We discretize again using implicit difference schemes backward in time and central space to v_3 and central space scheme to μ [16]. The algorithm remains stable, just as we wanted.

The Stable Fluids also includes a method to compute the motion of a density ρ immersed in the fluid. The equation for the evolution of this density is

$$\partial_t \rho = -(v \cdot \nabla)\rho + \kappa \triangle \rho + S$$

where κ is a diffusion rate and S is a source of density.

B. Fluid and Warping

The strategy is to move the coordinates of a parametrization of the image through the fluid velocity field. In this way, the image features are preserved by a texture mapping mechanism. We regard a fluid in the image domain and move this fluid taking external forces as initial data. For a point (x, y) in the image domain, each coordinate x and y is interpreted as a density immersed in the fluid on domain. We move each coordinate separately through the fluid velocity field, by equation 1 or 2 getting for a time step $\triangle t$ the coordinates w and z. Finally, the value of the new image at point (x, y) is the value of the original image at (w, z). For more details see [14].

C. Adjoint Method

In this paper we adopt of concept the adjoint method for a precise control of the fluid simulation. The adjoint method is an efficient and effective technique for calculating the gradient of a real valued function. For this the method computes the gradient directly form the numerical code of the simulation, following the general rule: we can express the matrix product

$$Q_2 = A_2 Q_1$$

as the product of a matrix by a vector

$$p_1 = (A_2)^\top p_2$$

each derived from the code instruction. For more details, see [12] and [17].

III. Control Techniques

If the animators desire to *control* a fluid warping, then this meaning they have to control the fluid simulation. Perhaps one would like a object in a image to follow a certain shape at a specific time. To achieve this, the users would think that to control the simulation they can apply forces in the flow, however is more than that. Here we developed three control techniques: Force and Viscosity, Variable Viscosity and Particle Keyframes. Now we present the general idea of each control technique and in the next section we will see the input parameters that the users have to manipulate each technique.

A. Force and Viscosity

Viscosity is a measure of the resistance of a fluid which is being deformed by either shear stress or tensile stress. Thus, the water have a lower viscosity, while honey have a higher viscosity. Then, less viscous the fluid is, the greater its ease of movement. An user might try to control the flow by applying external forces throughout the simulation carrying the image towards its goal. But for a low viscosity the non-linear nature of fluid movement make tiny change to forces in the simulation capable of large and unexpected changes in simulation states. Then for a intermediate viscosity the fluid has more resistance to be deformed by the forces. For high-viscosity the deformation is imperceptible. The viscosity is a important control parameter. Accordingly in this technique the user can to control the warping by viscosity and forces. See figure 2.

(a) visc = 0.001 (b) visc = 0.002 (c) visc = 0.005

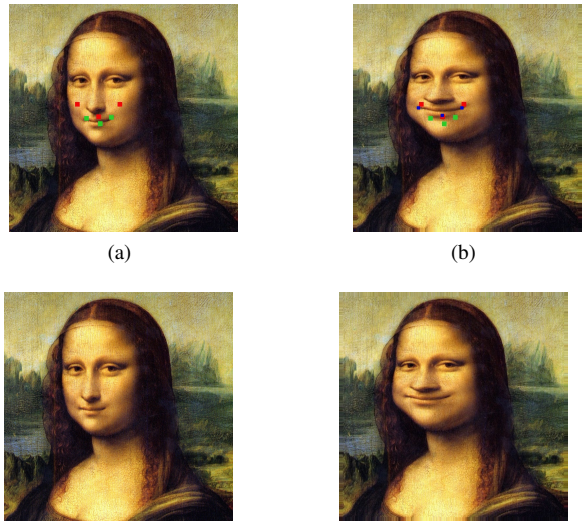
Fig. 2. For all of case here force is the same. In (a) for a low viscosity the simulation capable of unexpected and large changes. In (b) for a intermediate viscosity the fluid has more resistance to be deformed by the forces. In (c) for high-viscosity the deformation is imperceptible

B. Variable Viscosity

For this technique the user control the warping again using forces and viscosity. But here the viscosity is a spacial variable. This gives the possibility to specified regions in the image with more or less deformation. In this way, we specified low viscosity in the regions where we desired more movement or deformation, and then, we define high viscosity where the deformation must be less. For this definition we associate the viscosity with the image intensity. The user can apply forces associated with features of the image.

C. Particle-Keyframes

In this technique the user specifies two points in the image. The first one, which is to deformation and we call it *source point* p, and the second point k which we call *destination point* that p should match together as closely as possible at time t. Then the set of positions of source points is our simulations states q and the set of destination points k at time t_* is a keyframe K. See figure 3. The user specifies a keyframe K that the state q_t should achieve. To influence the simulation our system also needs a set of forces. To find this forces is automatic task.



(a) (b)

(c) Original Image (d) Final Result

Fig. 3. Control by Particle-Keyframe. In (a) Green points are leave points, and the set of red points are the keyframe In (b) blue points are the simulation state at the final time T, in our case final time is the keyframe time t_*.

Before of the beginning of the automatic process, the user specifies the viscosity.
The forces have specific positions and times (when they should be applied to influence the simulation).

The forces may come from a model and have specific positions but, we don't know the each magnitude and directions. For calculate the magnitude and direction of the forces, first we encapsulate all the forces in a vector $\mathbf{u}$. In $\mathbf{u}$ we include the forces magnitude, direction and postions. Any given value of $\mathbf{u}$ defines some unique set of forces applied to the velocity field and hence a unique simulation. Second, we

use an optimization process to get the control vector $\mathbf{u}$ which produces the simulation states that best match the keyframe.

1) Objective Function: We must be able to measure how well the simulation states match the keyframes. For this we define real-valued function called objective function as follow:

$$\varphi(\mathbf{u}) = \frac{1}{2} \sum_{t \in C_p} \left(\| \left(\mathbf{q}_t - \mathbf{K}_t \right) \|^2 \right) \tag{3}$$

Where C_p is a set of timesteps with positions keyframes. Similar functions have already been defined in [10] and [18]. Here we designed a different function, because we measure distances between points, and thus the function has a unique minimum, while in the case [10] and [18] the function can exhibit local minimums. Therefore to resolve this issue, they have to blur both the state and keyframe before evaluation which is not the ideal.

2) Optimization Process and Adjoint Method: As Treuille *et al.* [10] and McNamara *et al.* [18], we use a limited memory quasi-Newton optimization technique [19], which approximates the second derivative (the Hessian matrix) from a small set of most recent gradients. The quasi-Newton optimization requires us to evaluate φ and its gradient $\frac{\partial \varphi}{\partial u}$ for any value of the control parameters $\mathbf{u}$. For this task we use the adjoint method to compute efficiently the gradient of objective function.

IV. Control Parameters

For each previous technique there are a set of control parameters that allow to influence and to control the simulation, here we present possible control parameters for each method.

A. Parameters of Viscosity and Force

For this technique the control parameters are the viscosity and forces. The viscosity is a important tool to control the warping. The relevant aspects in the forces are the position, magnitude, direction and time, when and where they are must be apply. They can also be grouped or not. In relation to time the forces can be classified the following way: 1) instantaneous forces (i.e., acting at specific time instants t_i, $i = 1, ..., N$); 2) constant forces (i.e., acting during a certain time interval $[t0, t1]$; 3) arbitrary forces (i.e., fully variable in time). We found that the most interesting control parameter by its result in animation of images is the *oscillator force*.

Oscillator Force: Suppose that we have a basic force f_w, scaled by an oscillatory function, creating a oscillator force. We use this force to create animations. Our force is then

$$F = f_w \cdot f_{osc}$$

where

$$\mathbf{f}_{osc}(t) = A \; sen(tk)$$

with amplitude A, period T and frequency $k = 1/T$, the number of cycles per unit time. See figure 4

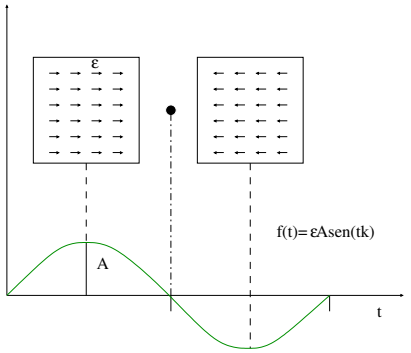


Fig. 4. Oscillator Force

B. Parameters of Variable Viscosity

By the Navier-Stokes equations 2 for variable viscosity is easy to verify that the available parameters are:

- external forces $f(x, t)$; and
- fluid viscosity $\mu(x)$.

An extra parameter is the total duration T of the simulation. Both the forces and viscosity are spatially variable. Thus, they are identified with functions on the image domain which could be associated with image features. Therefore, one way to specify forces and viscosity is by auxiliary images. We define the viscosity from the intensity of an auxiliary image function. The viscosity values are computed from a normalization of the image values to $[0, 1]$ and global scaling factor.

V. Results

We made several experiments applying the techniques above showed, to create animations and images morphing. In this section we give some examples of the results that can be obtained with fluid warping and its control methods.

The first set of examples demonstrate the use of oscillator forces to control image animations.
In the Figure 5 we have a image of Oliver Hardy and we create an animation from the image. For this we used a Gaussian forces (see [10]) with diagonal direction over the nose and we used the osillator force to create the animation. Is a real time animation with 15 frames per second. In the same way we have Figure 6, but here the animation has vertical direction force.

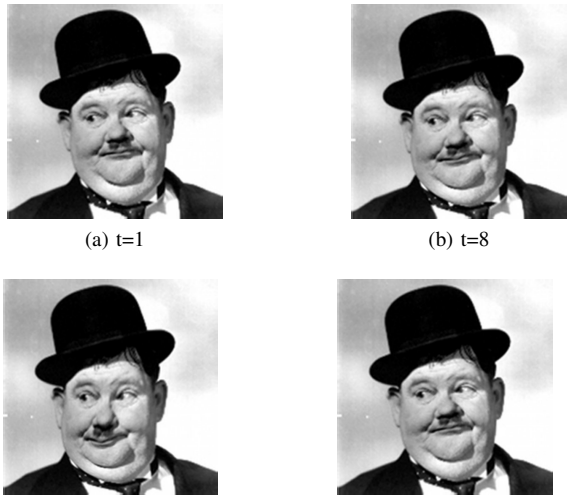
The last figure 7 show an example of the use of particle-keyframes method to create a morphing sequence.

VI. Conclusion

In this paper, image warping using fluid dynamics is presented. Authors present the fluid simulation setting, including the mathematical formulation and the Stable Fluids algorithm is adapted for variable viscosity. We explain the parameters used for controlling the image warping.

References

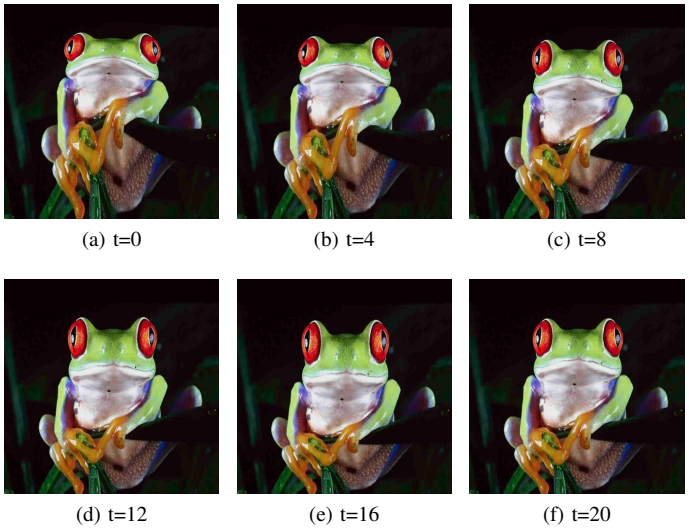
[1] J. Gomes, L. Darsa, B. Costa, and L. Velho, *Warping and Morphing of Graphical Objects*. Morgan Kaufmann Publ., 1999.



(a) t=1 (b) t=8

(c) t=11 (d) t=17

Fig. 5. Here an animation of a picture of Oliver Hardy. We move the mouth and nose with a oscillator force in the diagonal direction, centered on the mouth.



(a) t=0 (b) t=4 (c) t=8

(d) t=12 (e) t=16 (f) t=20

Fig. 6. Frog Animation. From image we create an animation, using oscillator force applied on the fluid defined on the domain image.

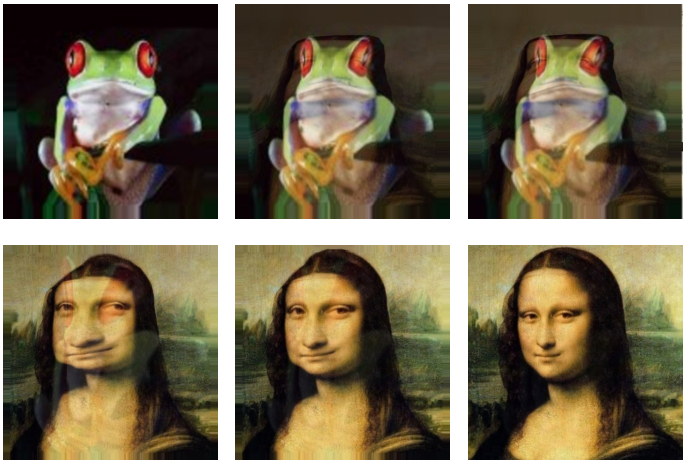


Fig. 7. Morphing sequence.

[2] A. H. Barr, "Global and local deformations of solid primitives." *In Proceedings of SIGGRAPH*, 1984.

[3] N. Arad and D. Reisfeld, "Image warping using few anchor points and radial functions," *Computer Graphics Forum*, vol. 14, no. 1, pp. 35–46, 1995.

[4] A. R. Smith, "Planar 2-pass texture mapping and warping," *In Proceedings of SIGGRAPH*, 1987.

[5] N. Foster and D. Metaxas, "Realistic animation of liquids," *Graph. Models Image Process.*, vol. 58, no. 5, pp. 471–483, 1996.

[6] ——, "Modeling the motion of a hot, turbulent gas," in *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1997, pp. 181–188.

[7] J. Stam, "Stable fluids," *SIGGRAPH 99 Conference Proceedings, Annual Conference Series*, pp. 121–128, August 1999.

[8] N. Foster and R. Fedkiw, "Practical animation of liquids," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2001, pp. 23–30.

[9] D. Enright, S. Marschner, and R. Fedkiw, "Animation and rendering of complex water surfaces," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 736–744, 2002.

[10] A. Treuille, A. McNamara, Z. Popović, and J. Stam, "Keyframe control of smoke simulations," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 716–723, 2003.

[11] M. B. Giles and N. A. Pierce, "An introduction to the adjoint approach to design," *Flow, Turbulence and Combustion*, vol. 65, pp. 393–415, 2000.

[12] R. Giering and T. Kaminski, "Recipes for adjoint code construction," *ACM Transactions on Mathematical Software*, vol. 24, no. 4, pp. 437–474, 1998.

[13] M. Bertalmio, A. Bertozzi, and G. Sapiro, "Navier-stokes fluid dynamics and image and video inpainting." *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, pp. 9–14, December 2001.

[14] D. Bonilla, L. Velho, A. Nachbin, and L. Nonato, "Fluid warping," in *Proceedings of the IV Iberoamerican Symposium in Computer Graphics*, O. Rodríguez, F. Serón, R. Joan-Arinyo, and J. R. E. C. J. Madeiras, Eds., Sociedad Venezolana de Computación Gráfica. DJ Editores, C.A., June 2009.

[15] R. Courant, E. Isaacson, and M. Rees, "On the solution of nonlinear hyperbolic differential equations by finite differences," *Communications on Pure and Applied Mathematics*, vol. 5, pp. 243–255, 1953.

[16] J. C. Strikwerda, *Finite Difference Schemes and Partial Differential Equations*. CRC Press, 1999.

[17] J. Stam, "Simulation and control of physical phenomena in computer graphics," in *PG '04: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 171–173.

[18] A. McNamara, A. Treuille, Z. Popović, and J. Stam, "Fluid control using the adjoint method," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 449–456, 2004.

[19] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, 1997.

[20] C. Wojtan, P. J. Mucha, and G. Turk, "Keyframe control of complex particle systems using the adjoint method," in *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 15–23.

[21] N. Okazaki, "libLBFGS: a library of Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS)," http://www.chokkan.org/software/liblbfgs/, 2009, [Online; accessed 22-January-2010].

[22] M. Bücker, "Community Portal for Automatic Differentiation," http://www.autodiff.org/, 2000, [Online; accessed 22-January-2010].

[23] H. Birkholz and D. Jackèl, "Image warping with feature curves," *In Proceedings of SIGGRAPH*, pp. 199–202, 2003.

[24] A. Chorin and J. E. Marsden, *A mathematical Introduction to Fluid Mechanics*. Springer-Verlag, 1993.

[25] J. Gomes and L. Velho, *Image Processing for Computer Graphics*. Springer Verlag, 1997.

[26] P. S. Heckbert, *Fundamentals of Texture Mapping and Image Warping*. University of California, Berkeley: Master's Thesis, 1989.

[27] J. Stam, "Flows on surfaces of arbitrary topology," *ACM Transactions On Graphics (TOG), Proceedings of SIGGRAPH*, pp. 724–731, July 2003.

[28] T. Beier and S. Neely, "Feature-based image metamorphosis." *SIGGRAPH Comput.*, vol. 2, no. 26, pp. 35–42, July 1992.

[29] S.-Y. Lee, G. Wolberg, Kyung-YongChwa, and S. Y. Shin, "Image metamorphosis with scattered feature constraints," *IEEE Transactions on Visualization and Computer Graphics*, vol. 2, no. 4, pp. 337–354, December 1996.

[30] A. Nachbin, *Notas do Curso: Dinâmica dos Fluidos*, 2006.

[31] S. Schaefer., McPhail.T, and J. Warren., "Image deformation using moving least squares," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, July 2006.

[32] D. B. Smithe, "A two-pass mesh warping algorithm for object transformation and image interpolation." *Technical memo, Industrial Light and Magic*, 1990.

[33] G. Wolberg, "Skeleton based image warping," *The Visual Computer*, vol. 5, no. 1-2, pp. 95–108, January 1989.

[34] ——, *Digital Image Warping*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1990.

[35] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual simulation of smoke," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2001, pp. 15–22.

[36] Y. Zhu and R. Bridson, "Animating sand as a fluid," in *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*. New York, NY, USA: ACM, 2005, pp. 965–972.

[37] M. Müller, D. Charypar, and M. Gross, "Particle-based fluid simulation for interactive applications," in *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003, pp. 154–159.

[38] R. Keiser, B. Adams, D. Gasser, P. Bazzi, P. Dutre, and M. Gross, "A unified lagrangian approach to solid-fluid animation," *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics*, vol. 0, pp. 125–148, 2005.

[39] B. Adams, M. Pauly, R. Keiser, and L. J. Guibas, "Adaptively sampled particle fluids," in *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*. New York, NY, USA: ACM, 2007, p. 48.

[40] C. Wojtan and G. Turk, "Fast viscoelastic behavior with thin features," in *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*. New York, NY, USA: ACM, 2008, pp. 1–8.

[41] A. W. Bargteil, C. Wojtan, J. K. Hodgins, and G. Turk, "A finite element method for animating large viscoplastic flow," in *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*. New York, NY, USA: ACM, 2007, p. 16.

[42] M. Carlson, P. J. Mucha, and G. Turk, "Rigid fluid: animating the interplay between rigid bodies and fluid," in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*. New York, NY, USA: ACM, 2004, pp. 377–384.

[43] N. Kwatra, C. Wojtan, M. Carlson, I. Essa, P. J. Mucha, and G. Turk, "Fluid simulation with articulated bodies," *IEEE Transactions on Visualization and Computer Graphics (TVCG 2010)*, 2010.

[44] R. Fattal and D. Lischinski, "Target-driven smoke animation," in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*. New York, NY, USA: ACM, 2004, pp. 441–448.