Pattern Analysis and Applications manuscript No.
(will be inserted by the editor)

Motion synthesis through 1D affine matching

Perfilino E. Ferreira Jr[1,2], José R.A. Torreão[3], Paulo Cezar Pinto Carvalho[1], Marcelo Bernardes Vieira[1]

[1] Instituto Nacional de Matemática Pura e Aplicada – Estr. Dona Castorina, 110, 22460-320 Rio de Janeiro, RJ, Brazil
[2] Universidade Federal da Bahia – Av. Ademar de Barros, s/n, 40170-110 Salvador, BA, Brazil
[3] Universidade Federal Fluminense – R. Passo da Pátria, 156, 24210-240 Niterói, RJ, Brazil

Received: March 15, 2006 / Revised version: May 20, 2007

Abstract We present the study of a data-driven motion synthesis approach based on a 1D affine image-matching equation. We start by deriving the relevant properties of the exact matching operator, such as the existence of a singular point. Next, we approximate such operator by the Green's function of a second-order differential equation, finding that it leads to a more compelling motion impression, due to the incorporation of blur. We then proceed to show that, by judicious choice of the matching parameters, the 1D affine Green's filter allows the simulation of a broad class of effects, such as zoom-in and zoom-out, and of complex nonrigid motions such as that of a pulsating heart.

Key words Motion synthesis, matching equation, Green's functions

1 Introduction

Data-driven motion simulation is a relatively unexplored subject in computer graphics, since most of the motion simulation algorithms found in the literature are model-based ones. In [1] and [2], for instance, stochastic models are used, the latter being based on an *autoregressive process*. A deterministic model is that of [3], where the motion of fluids is simulated through the Navier-Stokes equation. On the other hand, a fully data-driven approach is that of Freeman et al., who use *steerable filters* [4] to create their 'motion without movement' illusion [5].

An important issue related to motion simulation is that of lending physical realism to the generated sequences, what usually requires the introduction of motion blur [6]-[9] as an independent step in the process. The simultaneous synthesis of general motion and motion blur does not seem to have been undertaken. Here we pursue this, by considering solutions of matching (irradiance-conservation) equations of the form

$$f_2(x+U, y+V) = f_1(x, y), \tag{1}$$

where f_1 is the input image, $U(x, y)$ and $V(x, y)$ are the optical flow components [10]-[12], and f_2 is a matching image, to be found, which, along with f_1, will convey the motion information.

Approximate solutions to equation (1), for a uniform optical flow field — $U(x, y) = u,\ V(x, y) = v$ — have already been considered in [13], in the context of 3D shape estimation. In this case, expanding the left-hand side in a Taylor series up to second order in u and v, and performing a suitable change of variables, equation (1), for matching along a general direction θ — i.e., for $v/u = \tan\theta$ — reduces to the one-dimensional form [13]

$$\frac{u^2}{2} f_2{}'' + u f_2{}' + f_2 = f_1, \tag{2}$$

where $f_i = f_i(x, y + \gamma x)$, with $\gamma = \tan\theta$, and where the primes denote differentiation with respect to x.

Through the Green's function approach, a solution to (2) can then be found as

$$f_2(x, y+\gamma x) = \int_{\mathcal{D}} G_u(x-\xi) f_1(\xi, y+\gamma\xi) d\xi, \tag{3}$$

where G_u is a linear, shift-invariant filter which is the Green's function to equation (2) - that is to say, it is the solution to that equation when the unit impulse function $\delta(x-\xi)$ is substituted for its right-hand side. Over an infinite domain $\mathcal{D}$, G_u will take the form

$$G_u(x-\xi) = \frac{2}{u} \sin\left(\frac{x-\xi}{u}\right) e^{-\left(\frac{x-\xi}{u}\right)}, \tag{4}$$

for $x > \xi$, with $G_u = 0$, otherwise.

As shown in [13], when convolved with a shading image, G_u is able to simulate its *photometric stereo* pair - i.e., a rendition of the same scene under displaced illumination. It thus allows the purely data-driven simulation of a certain kind of 3D motion - namely, that of the irradiance pattern over a static scene, arising from a change in illumination direction (as proven in [14], this can be generally modeled as a non-uniform

rotation). It should be noted that the Green's filter G_u is comprised of two parts: a sinusoidal function and an exponential function. Considered by itself, the sinusoidal factor would essentially induce a displacement of the image intensities, what makes it account for most of the motion effect of the Green's filter. The exponential term, on the other hand, induces essentially a blurring effect. This can be seen by introducing G_u in equation (3), and rewriting it as

$$f_2(x,y) = \frac{2}{u}\int_{-\infty}^{\infty}\left[\sin\left(\frac{\xi}{u}\right)S(\xi)\right]e^{-\frac{|\xi|}{u}}f_1(x-\xi,y)d\xi, \tag{5}$$

where S denotes the unit step function, and where, for simplicity, we considered the case when $\gamma = 0$. By itself, the negative exponential term in (5) is simply a blurring factor, its Fourier transform being $2u/(1+u^2\omega^2)$. On the other hand, the term with the sine function carries, in its Fourier spectrum, contributions of the form $\delta(\omega \pm 1/u)$, whose effect will be to select the frequency component $1/u$ from the input signal f_1, thus producing a sinusoidal spreading of its intensities. Both kinds of effect combine in G_u to produce a shift-invariant point spread function that simultaneously models both motion and motion blur.

Here we present a way of extending the Green's function approach described above, aiming at the simulation of a broader class of movements. We do this by considering a less restrictive optical flow model: instead of the uniform field which led to (4), we introduce the affine model $U(x) = u_0 + u_1 x$, where both u_0 and u_1 are constants (again, we assume one-dimensional flow, with $V(x) = \gamma U(x)$). Considering the solution, under this new model, to the approximate matching equation equivalent to (2), we obtain a Green's function filter which, when applied to a single input, is able to generate image sequences that simulate various kinds of uniform and nonuniform motions. Similar to the uniform flow case, the advantage of the affine Green's function approach is that it allows the generation of motion blur simultaneously with motion synthesis, what contributes to the greater realism of the result.

Two other algorithms that make use of 1D affine transformations to generate motion and motion blur effects are the ones presented in [15] and [16]. The former is a two-step process, that first warps and then blurs the input image. The latter generates motion and blur simultaneously, but is specific for a certain kind of motion, namely, a zoom-in (its code is a plug-in for the GIMP software, available free of charge from [17]). Here, we show that the Green's function model compares favorably with both these approaches.

The remainder of this paper is organized as follows: in Section 2, we consider the one-dimensional affine matching equation and the Green's function solution to an approximate version of it, also discussing some features of its possible extension to 2D; in Section 3, we consider the practical implementation of the affine Green's filter; in Section 4, we present and discuss some experimental results, including a comparison with two motion-blur simulation approaches; finally, in Section 5, we make our concluding remarks and propose directions for future work, and in Section 6, we present an overview of our contributions.

2 One-dimensional Affine Matching

Here we are concerned with a one-dimensional matching equation of the form

$$f_2(x+U) = f_1(x) \tag{6}$$

for an affine optical flow field, $U(x) = u_0 + u_1 x$, where both u_0 and u_1 are constants which can be interpreted, respectively, as the optical flow value at $x = 0$, and its derivative. We first derive some general properties of the matching equation, and then proceed to consider its approximate solution via the Green's function approach. Although we here assume matching along the direction x, generalization of our results to a general matching direction — i.e., when f_1 and f_2 are functions of $(x, y + \gamma x)$, as in (2) — is straightforward.

2.1 Properties of the Affine Matching Transformation

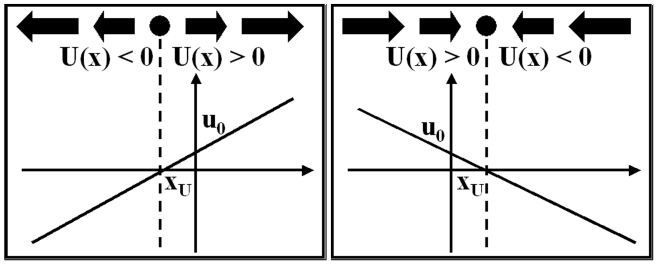
Matching equation (6) can be rewritten as

$$f_2(x) = M_U\left[f_1\right](x), \tag{7}$$

where M_U is the linear mapping

$$M_U\left[f\right](x) = f\left(\frac{x-u_0}{1+u_1}\right). \tag{8}$$

We start by noting that, whenever $u_1 \neq 0$, the affine field $U(x)$ presents a zero at $x_U = -\frac{u_0}{u_1}$, corresponding to a singular point (fixed point) of the mapping, for which $f_2(x) = f_1(x)$. From the form of (8), it is also easy to see that, except at $x = x_U$, the mapping will consist of a translation by $\frac{u_0}{1+u_1}$ and a scaling by $\frac{1}{1+u_1}$. The latter amounts to an expansion, when $u_1 > 0$, and to a contraction, when $u_1 < 0$. The net effect of the combined translation and scale transformations will be that of a displacement away from the fixed point, for positive u_1 (Fig. 1a), and towards the fixed point, when u_1 is negative (Fig. 1b), the displacement increasing linearly with the distance from the fixed point. Such features are illustrated by Fig. 2, where we present the affine mapping of the function $f_1(x) = \frac{\sin(x^2)}{x}$.



(a) (b)

Fig. 1 Illustrating the effect of the affine matching operator: (a) Expansion ($u_1 > 0$). (b) Contraction ($u_1 < 0$).

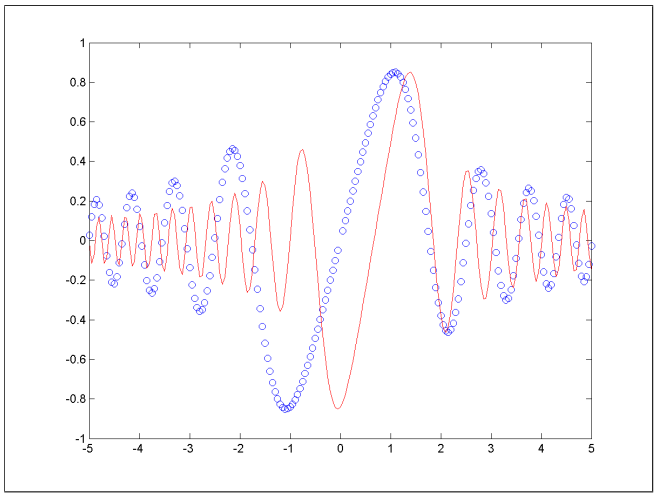


Fig. 2 Application of the operator M_U, for $(u_0, u_1) = (\frac{2}{3}, -\frac{1}{3})$, to the function $f_1(x) = \frac{\sin(x^2)}{x}$ (circles). The resulting signal, $f_2(x) = M_U[f_1](x) = \frac{\sin\left(\frac{x-u_0}{1+u_1}\right)^2}{\left(\frac{x-u_0}{1+u_1}\right)}$, is shown as a continuous line.

2.2 Green's Function Solution

Let us now consider the Green's function solution to an approximate version of the affine matching equation, namely, that given by the second-order Taylor-series expansion of the left-hand side of (6):

$$\frac{(u_0 + u_1 x)^2}{2} f_2'' + (u_0 + u_1 x) f_2' + f_2 = f_1. \quad (9)$$

Equation (9) has the Cauchy-Euler form [18], and its Green's function solution, over a domain $\mathcal{D}$, will be expressed as

$$f_2(x) = \int_{\mathcal{D}} G(x, \xi) f_1(\xi) d\xi, \quad (10)$$

where $G(x, \xi)$ is the *shift-variant* Green's function which solves

$$\frac{(u_0 + u_1 x)^2}{2} G'' + (u_0 + u_1 x) G' + G = \delta(x - \xi) \quad (11)$$

over that same domain. Here, the notation $f_1 * G$ will be used as a shorthand for the operation on the right-hand side of (10).

Under the sole condition that $G(x, \xi)$ remains finite as x goes to infinity, a solution to (11) can be found as

$$G_+(x, \xi) = \frac{2}{{u_1}^2 \beta (\xi - x_U)} \left[\frac{x - x_U}{\xi - x_U}\right]^\alpha \sin\left\{\beta \log\left[\frac{x - x_U}{\xi - x_U}\right]\right\}, \quad (12)$$

for $x > \xi$, with $G_+(x, \xi) = 0$, otherwise. The parameters α and β are given as

$$\begin{cases} \alpha = -\frac{1}{u_1} + \frac{1}{2} \\ \beta = \frac{1}{u_1}\sqrt{1 + u_1 - \frac{u_1^2}{4}}, \end{cases} \quad (13)$$

leading to a bounded Green's function over a domain $\mathcal{D} \subset (x_U, \infty)$, so long as we take $0 < u_1 < 2$. Over finite domains, this solution is valid for $2 - 2\sqrt{2} < u_1 < 2 + 2\sqrt{2}$. A plot of the G_+ filter appears in Figure 3.

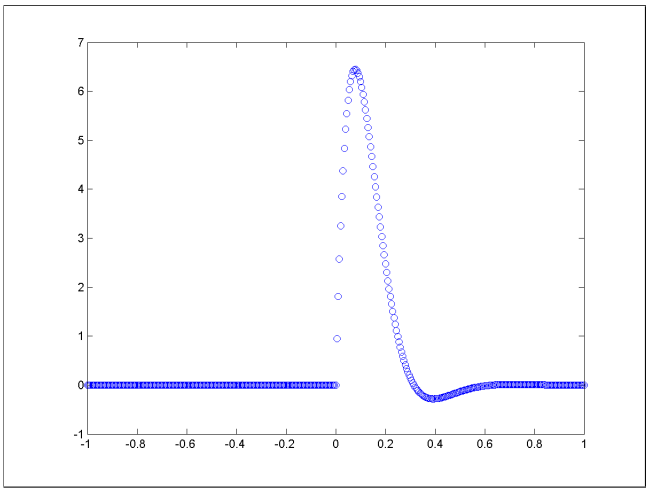


Fig. 3 G_+ filter plotted as function of x, for $\xi = 0$. Parameter values: $u_0 = 0.1$ and $u_1 = 0.002$.

Over a domain $\mathcal{D} \subset (-\infty, x_U)$, a solution to (11) can also be found as

$$G_-(x, \xi) = \frac{2}{{u_1}^2 \beta (x_U - \xi)} \left[\frac{x_U - x}{x_U - \xi}\right]^\alpha \sin\left\{\beta \log\left[\frac{x_U - x}{x_U - \xi}\right]\right\}, \quad (14)$$

for $x < \xi$, with $G_-(x, \xi) = 0$, otherwise. In this case, the parameters α and β become

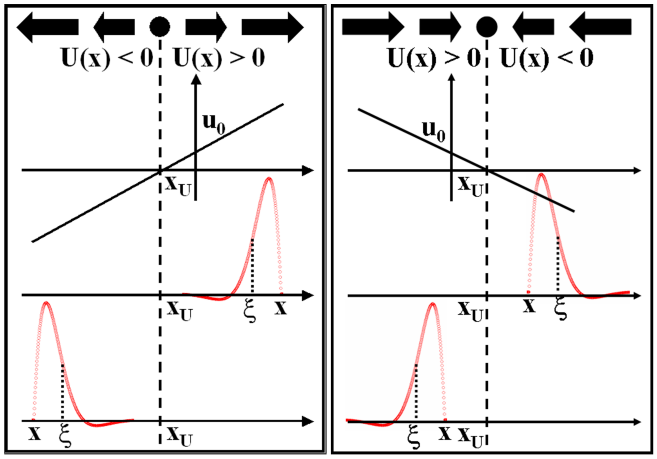
$$\begin{cases} \alpha = \frac{1}{u_1} + \frac{1}{2} \\ \beta = \frac{1}{u_1}\sqrt{1 - u_1 - \frac{u_1^2}{4}}, \end{cases} \quad (15)$$

and G_- will be bounded so long as we take $-2 < u_1 < 0$. In finite domains, this solution is valid for $-2 - 2\sqrt{2} < u_1 < -2 + 2\sqrt{2}$.

Similar remarks as made for the uniform-flow Green's filter G_u, of equation (4), apply to the forms G_+ and G_- above: both can be factorized into two terms, one of them (the sinusoidal factor) basically inducing the displacement of image intensities, and the other introducing image blur. Once again, we are dealing with

point spread functions (shift-variant, in this case) that simultaneously model motion and motion blur processes.

An illustration of the roles of the G_+ and G_- filters, in the approximation of affine matching, is presented in Figure 4. For instance, in the generation of expansion, illustrated by Fig. 4(a), the value of f_2 at a point x in the range $(x_U, +\infty)$ will depend on the values of $f_1(\xi)$ for every $\xi < x$, each of them weighted by the corresponding Green's function value $G_+(x, \xi)$, according to equation (10). Similarly, at each point in the $(-\infty, x_U)$ range, the values of $f_1(\xi)$ for every $\xi > x$ will be weighted by $G_-(x, \xi)$, in order to produce $f_2(x)$. The analysis of the contraction case (Fig. 4(b)) can be similarly performed.



(a) (b)

Fig. 4 Illustrating the roles of the G_+ and G_- filters in motion simulation. (a) Expansion, with $u_0 > 0$ and $u_1 > 0$: The central plot shows $G_+(x, \xi)$, and the lower plot shows $G_-(x, \xi)$, as functions of ξ, for a fixed x. (b) Contraction, with $u_0 > 0$ and $u_1 < 0$: Similarly as in (a), but with G_- in the central plot and G_+ in the lower one (see text).

Fig. 5 illustrates the effect of the Green's function filter over the same signal as considered in Fig. 2. The input signal, $f_1(x)$ (circles), and its affine mapping $M_U[f_1](x)$ (continuous line) are plotted along with the curve for $f_1 * G$ (crosses), obtained as presented in Section 3, below. In the illustrated case, $u_0 = \frac{2}{3}$ and $u_1 = -\frac{1}{3}$, what leads to a contraction and to a fixed point at $x_U = 2$. We note that the curves of f_1, $M_U[f_1]$ and $f_1 * G$ coincide at the fixed point, as expected.

2.3 Extension to 2D Affine Matching

Under the two-dimensional affine model, the optical flow components U and V, in equation (1), take the form

$$\begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} + \begin{bmatrix} u_1 & u_2 \\ v_1 & v_2 \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{b} + \mathcal{A}.\mathbf{X}, \qquad (16)$$

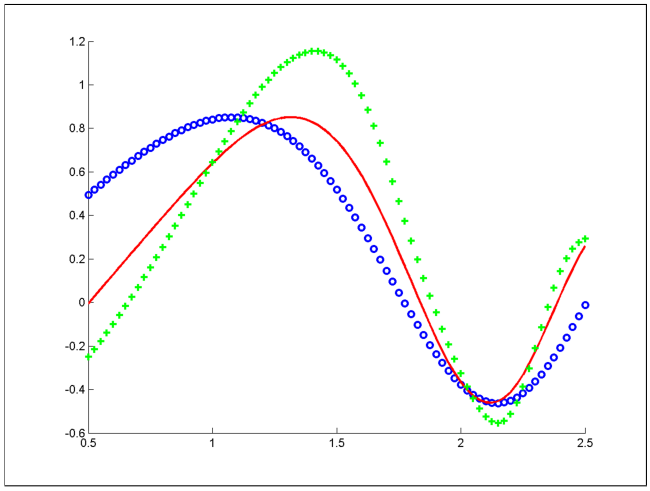


Fig. 5 Plots of $f_1(x) = \frac{\sin(x^2)}{x}$ (circles), $f_2(x) = M_U[f_1](x) = \frac{\sin\left(\frac{x-u_0}{1+u_1}\right)^2}{\left(\frac{x-u_0}{1+u_1}\right)}$ (continuous line), and $f_1 * G$ (crosses), for parameters $u_0 = \frac{2}{3}$ and $u_1 = -\frac{1}{3}$.

with

$$\begin{cases} \mathbf{b} = [u_0 \ v_0]^T \\ \mathcal{A} = \begin{bmatrix} u_1 & u_2 \\ v_1 & v_2 \end{bmatrix} \\ \mathbf{X} = [x \ y]^T \\ u_i \text{ and } v_i \text{ constant } (i = 0, 1, 2) \end{cases} \qquad (17)$$

Performing a Taylor-series expansion up to second order, as in the 1D case, we would obtain, for the matching equation,

$$\begin{aligned} &\frac{U^2}{2}\frac{\partial^2 f_2}{\partial x^2} + UV\frac{\partial^2 f_2}{\partial x \partial y} + \frac{V^2}{2}\frac{\partial^2 f_2}{\partial y^2} + U\frac{\partial f_2}{\partial x} \\ &+V\frac{\partial f_2}{\partial y} + f_2 = f_1, \end{aligned} \qquad (18)$$

with $(U, V)^T = (U(x, y), V(x, y))^T$ as above. Its Green's function solution would then be expressed as

$$f_2(x, y) = \int\int_{\mathcal{D}} G(x, y; \xi, \eta) f_1(\xi, \eta) d\xi d\eta, \qquad (19)$$

where $\mathcal{D}$ is an appropriate domain. Equation (18) is of parabolic type [19]. In the case of uniform optical flow, $\mathcal{A} = \left[\begin{smallmatrix} 0 & 0 \\ 0 & 0 \end{smallmatrix}\right]$, it will reduce to

$$\begin{aligned} &\frac{u_0^2}{2}\frac{\partial^2 f_2}{\partial x^2} + u_0 v_0 \frac{\partial^2 f_2}{\partial x \partial y} + \frac{v_0^2}{2}\frac{\partial^2 f_2}{\partial y^2} + u_0\frac{\partial f_2}{\partial x} \\ &+v_0\frac{\partial f_2}{\partial y} + f_2 = f_1, \end{aligned} \qquad (20)$$

The above can be converted to the canonical form [19]:

$$\frac{u_0^2}{2}\frac{\partial^2 f_2}{\partial \sigma^2} + u_0\frac{\partial f_2}{\partial \sigma} + f_2 = f_1, \qquad (21)$$

where $f_i = f_i(\sigma, \zeta)$ $(i = 1, 2)$ and $(\sigma, \zeta) = [\sigma(x, y), \zeta(x, y)]$. Here, σ is a characteristic curve, that is to say, a curve along which the derivative is total [19]. In the present case, $(\sigma, \zeta) = [x, -v_0^2 x + u_0 v_0 y]$, and we note that equation (21) is identical to (2). Its Green's function will thus be similar to that in (4).

In the general case, a difficulty may arise when the reduction to the canonical form is not possible. One could, for instance, be able to obtain the transformation $(\sigma, \zeta) = [\sigma(x, y), \zeta(x, y)]$, but not the inverse $(x, y) = [x(\sigma, \zeta), y(\sigma, \zeta)]$. In the present work, we will not be concerned with the 2D affine model. We just note that, by employing the 1D model along both directions x and y, we are able to consider the equivalent to equation (18) in the particular case when

$$\frac{\partial^2 f_2}{\partial x \partial y} = 0 \quad \text{and} \quad \mathcal{A} = \begin{bmatrix} u_1 & 0 \\ 0 & v_2 \end{bmatrix}.$$

In this case, (18) reduces to

$$\frac{U^2}{2}\frac{\partial^2 f_2}{\partial x^2} + \frac{V^2}{2}\frac{\partial^2 f_2}{\partial y^2} + U\frac{\partial f_2}{\partial x} + V\frac{\partial f_2}{\partial y} + f_2 = f_1, \quad (22)$$

where $(U(x, y), V(x, y))^T = (u_0 + u_1 x,\ v_0 + v_2 y)^T$.

The Green's function of the above would be given by

$$G(x, y; \xi, \eta) = G_1(x, \xi).G_2(y, \eta),$$

where $G_1(x, \xi)$ and $G_2(y, \eta)$ are solutions to

$$\frac{(u_0 + u_1 x)^2}{2}\frac{d^2 G_1}{dx^2} + (u_0 + u_1 x)\frac{dG_1}{dx} + G_1 = \delta(x - \xi),$$

and

$$\frac{(v_0 + v_2 y)^2}{2}\frac{d^2 G_2}{dy^2} + (v_0 + v_2 y)\frac{dG_2}{dy} + G_2 = \delta(y - \eta),$$

respectively.

3 Implementation Issues

Here we discuss some practical implementation aspects of the affine Green's filter, as used for motion simulation. In the following, we only treat horizontal motion (i.e., $\theta = 0$), since motions in a general direction can be simulated by combining horizontal and vertical filtering, the latter being easily accomplished through the former, for a $\pi/2$-rotated input. Thus, we have only two independent filter parameters to consider, which we chose to be u_0 and x_U (keeping in mind that $u_1 = -u_0/x_U$). The first issue we wish to address regards the latter parameter, which is the position of the singularity point of the affine transformation.

We recall that the 1D Green's functions of equations (12) and (14) have been derived for the unbounded ranges $(x_U, +\infty)$ and $(-\infty, x_U)$, respectively. On the other hand, a practical domain for motion simulation will be a finite subset of $(-\infty, +\infty)$. Thus we have that, if x_U lies outside such domain, a single Green's function (G_+ or G_-) will be involved in the motion simulation, and if x_U lies inside it, one has to consider both G filters. The position of the singularity point relative to the image domain represents therefore an additional degree of freedom which can be exploited for the generation of a broader class of motions (we should mention here that the importance of affine fixed points for motion synthesis and analysis has long been recognized [20]-[25]).

Still regarding x_U, we should also remark that the Green's function is not defined at that point. Thus, in our simulations, whenever needed, we simply kept the original intensity value there. Points in the neighborhood of x_U, on the other hand, where G is defined and varies rapidly, will require oversampling. As a rule, area sampling should always be used, in order to minimize errors [26].

Some care is also required in the computation of $G(x, \xi)$, which corresponds to an integral evaluated at a pixel of the input image. A proper discretization procedure (we used the trapezoidal rule) is important to ensure unitary gain in the summation.

When performing both horizontal and vertical filtering, we are obviously free to choose independent Green's function parameters along the two directions. Thus, in the experiments described below, we also consider the values for v_0 and y_V, the latter being the fixed point along the vertical. This is defined, similarly to x_U, as $y_V = -\frac{v_0}{v_2}$, where v_2 is the rate of change of the optical flow along the $y-$direction (see Section 2, above).

Finally, we should comment that, in our implementations of the exact matching operator, linear interpolation was used whenever dealing with non-integer displacements.

4 Experimental Results

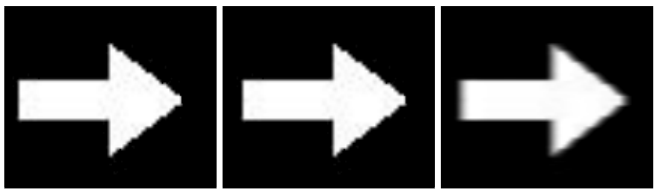
Animated motion sequences illustrating the application of our approach can be found at the web site http://www.graphics.ufba.br/FAPESB/motion/. Here we discuss the general features of such experiments. It should be noted that, as a rule, in the figures presented here, only a subset of the images comprising the animations have been included.

Given a single input image, its companions in each artificial motion sequence have been generated as described in Section 3 above, with x_U, y_V, u_0 and v_0 chosen in such a way as to produce the desired effects. As already mentioned, whenever simulating motion in a direction other than the horizontal, two filterings of the input image - along the horizontal and the vertical - were combined.

As a general rule, the motion sequences result more compelling when the values of u_0 are small, since the loss of high frequency information is less severe in this case. It should nevertheless be remarked that the blurring effect entailed by Green's function filtering is important in conveying a convincing motion impression to the human eye. In our experiments, we found that u_0 in the range of 1 to 8 pixels per frame yields good results, with the values up to 3 pixels/frame basically entailing negligible blur. The values for u_1 (which are obtained from those

for u_0 and the chosen x_U) are usually a fraction of those for u_0. Similar considerations hold as well for v_0, v_2 and y_V. Please note that, in the experiments reported here, the origin of the coordinate system is assumed to be at the image's bottom left corner, with positive coordinate axes running rightwards ($x-$axis) and up ($y-axis$).

We start by comparing motion simulations yielded by the approximate and the exact affine models. Fig. 6(b) was obtained by the direct application of the affine matching operator (8), and Fig. 6(c), by the integral operator of equation (10). It should be noted that the image in Fig. 6(b) is sharp, while there is loss of high frequencies in the Green's-function version of Fig. 6c. This happens because the image intensities in Fig. 6c are spreaded out, instead of simply displaced, as in Fig. 6(b). However, as already remarked, it is precisely this non-trivial blurring effect that helps to produce a more compelling motion impression. The role of motion blur in the creation of images that are more realistic and pleasing to the eye is well-known, and algorithms have long been developed to incorporate such effect in computer graphics [6]-[9]. With the Green's function approach, this comes along naturally with the motion synthesis process.



(a) (b) (c)

Fig. 6 Example of motion synthesis through affine matching (2D Translation). (a) Original image. (b) Motion synthesized via exact matching. (c) Motion synthesized via the $G-$filter. The parameters in both cases were $(u_0, u_1) = (4, 0.001)$

As a further example, Fig. 7 compares the direct and the Green's function approaches in the case of a zoom. Again we note that the synthesized image is sharp in the exact matching case, while the blur of the Green's function pair lends more realism to the image sequence, especially when simulating fast camera motions. Moreover, if the warping-pair images are presented to an observer in random order, he/she won't be able to tell which one was the original - that is to say, it will not be clear if the motion was a zoom-in or a zoom-out. In contrast, from the Green's-function pair, the kind of motion intended is immediately apparent, and motion cues can be inferred even from the second image alone, what could in principle allow the estimation of the direction and magnitude of the movement [27–29].

Next, still considering a zoom simulation, we compare the motion blur effects induced by the Green's function approach and by two other models, briefly described below.

Fig. 7 Example of motion synthesis through affine matching (Zoom): The second image of the pair on the left has been synthesized via exact matching, while the one on the right was generated via Green's functions. The parameters in both cases were $(u_0, u_1, x_U) = (-8, 0.125, 64)$ (horizontal motion) and $(v_0, v_2, y_V) = (-8, 0.125, 64)$ (vertical motion).

Motion Blur Model by Acha and Peleg [15]: In the Acha-Peleg model, the motion-blurred image is obtained by first warping the input, and then applying a space-invariant 1D blur kernel along the desired direction. Here, in order to simulate a zoom-in, we considered two 1D affine warpings followed by gaussian blur, along the horizontal and vertical directions.

Zoom Motion-Blur Model by Martinsen et al. [16]: This model is specific for zoom-in simulation. Here, an image pixel at position (x, y) is replaced by the average of those pixels located on a straight line starting at (x, y) and pointing towards the center pixel $(c_x, c_y) = (\frac{w}{2}, \frac{h}{2})$, where w and h are the image width and height, respectively. The blur length is a free parameter of the algorithm.

Figure 8 shows examples of zoom images generated with the above-described models, and with the Green's function approach. The images in Figs. 8(a) and (c), corresponding to the Acha-Peleg and Green's function models, respectively, have been generated with the same affine motion parameters, $(u_0, u_1, x_U) = (v_0, v_2, y_v) = (-4, 0.0625, 64)$. The image in Fig. 8(b), corresponding to the model by Martinsen et al., has been generated for $\gamma = 7$ and $(c_x, c_y) = (64, 64)$. Below each image, an enlarged portion of its upper right corner is presented, in Figs. 8(d)-(f).

From the images, it can be noted that the camera zoom effect, as produced by the Acha-Peleg model, is poor, mainly due to the uniform blur induced. The model by Martinsen et al., on the other hand, conveys a proper zoom impression, but introduces some artifacts, evident, for instance, over the woman's hat and on the background area (see Figs. 8(b) and (e)). In contrast, the Green's function image is free from artifacts, showing a much more even appearance. It also presents less blur over its central region, as should be expected, since the motion field would be very small there.

Similarly to Martinsen's, and contrary to Acha-Peleg's, the Green's function algorithm implements a

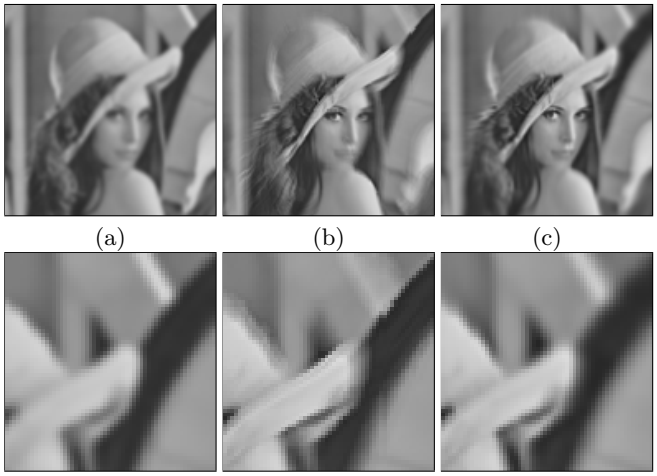


Fig. 8 Zoom motion blur: (a) Acha-Peleg model, for $(u_0, u_1, x_U) = (v_0, v_2, y_V) = (-4, 0.0625, 64)$, and gaussian standard deviation of 5 pixels. (b) Model by Martinsen et al., with center $(c_x, c_y) = (64, 64)$ and blur length $\gamma = 7$. (c) Green's function approach with $(u_0, u_1, x_U) = (v_0, v_2, y_V) = (-4, 0.0625, 64)$.

one-step motion/motion-blur simulation process, but, as will become clear from what follows, it has a much broader applicability, not being restricted to zoom simulation. Next, we will show how its parameters can be chosen, in order to synthesize different classes of motions.

The case when $u_1 = 0$, illustrated by Fig. 9, has already been considered in [13]. The corresponding Green's function is that of equation (4), which arises from a matching equation with constant coefficients. In this case, there is no singular point. A nonuniform rotation plus a translation can be inferred from the generated sequence. The impression of rotation is stronger when an increasing series of u_0 values is considered, e.g. $u_0 = 1, 2, ..., 8$, as in Fig. 9. We observed that the blurring of the object edges (in the present case, basically on the right-hand side) plays an important role in conveying a vivid rotation impression.

The 1D affine model introduces new simulation possibilities, through the additional parameter u_1. Below, we discuss these.

2D Translation *A***:** Can be obtained with affine parameters such that x_U lies outside the image domain. Here we illustrate the horizontal translation case, for u_0, $u_1 > 0$. A compelling translation impression is produced by sequences of increasing u_0 and u_1 values. For instance, the example in Fig. 10 was generated with $(u_0, u_1) = (1, 0.001)$, $(2, 0.002)$, $(3, 0.003)$, ..., $(7, 0.007)$. Since $U(x)$, the overall displacement, grows with x, at each frame the blur is stronger to the right. On the other hand, the increasing u_0 and u_1 values also lead to increasing blur with time.

2D Translation *B***:** This simulation is similar to the former case, but also combines horizontal and vertical

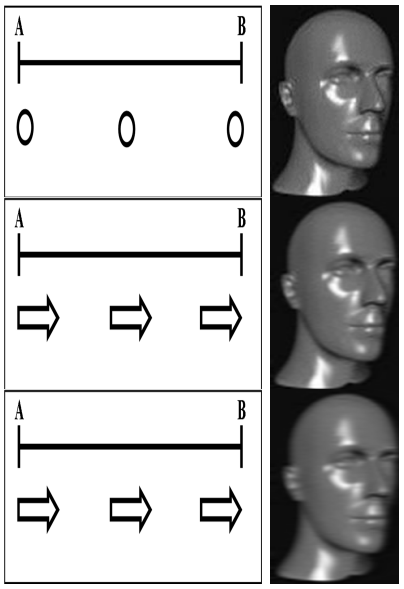


Fig. 9 Motion simulation with $u_1 = 0$ ($U(x) = u_0$). From top to bottom: input image and G_u-filtered sequence for parameter values $u_0 = 3$ and $u_0 = 6$, with $\theta = 0°$.

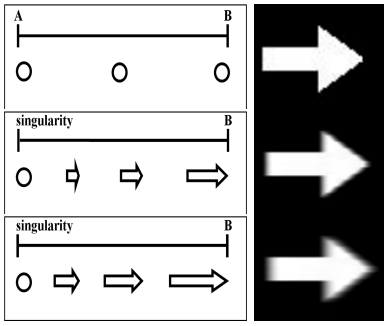


Fig. 10 2D Translation *A*. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1) = (4, 0.004)$, and $(7, 0.007)$. The singular point, $x_U = -1000$, lies outside the image.

filtering. The result is the motion of the pixels along straight lines running top-down and leftwards (Fig. 11). The parameter settings were $(u_0, u_1) = (-2, 0.008)$ and $(v_0, v_2) = (-1, 0.004)$, for the second frame; $(u_0, u_1) = (-4, 0.016)$ and $(v_0, v_2) = (-2, 0.008)$, for the third; and $(u_0, u_1) = (-8, 0.031)$ and $(v_0, v_2) = (-4, 0.016)$, for the fourth. The singular points lie outside the image.

The generation of more complex motions is illustrated by the examples below:

Pulsating Heart: In this case (Fig. 12), the singular point x_U lies at the center of the image, which is of 254×273 pixels. An expansion in the second frame, obtained with affine parameters $(u_0, u_1, x_U) = (-2, 0.015, 137)$ is followed by a contraction in the third, generated with $(2, -0.015, 137)$.

Zoom-Out: Here (Fig. 13), a superposition of filtering along the horizontal and vertical directions was performed. The affine parameters were $(u_0, u_1, x_U) = (v_0, v_2, y_V) = (2, -0.03, 64)$, for generating the second

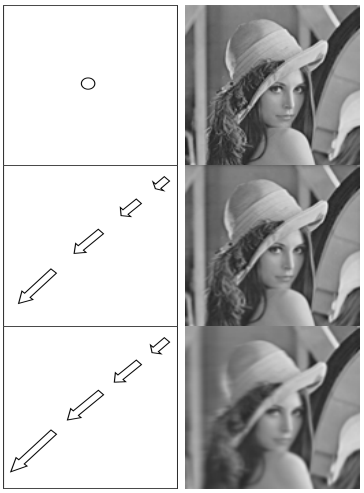


Fig. 11 2D Translation B. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1) = -2, 0.008)$ plus $(v_0, v_2) = (-1, 0.004)$, for the second frame; and $(u_0, u_1) = (-8, 0.031)$ plus $(v_0, v_2) = (-4, 0.016)$, for the fourth frame.

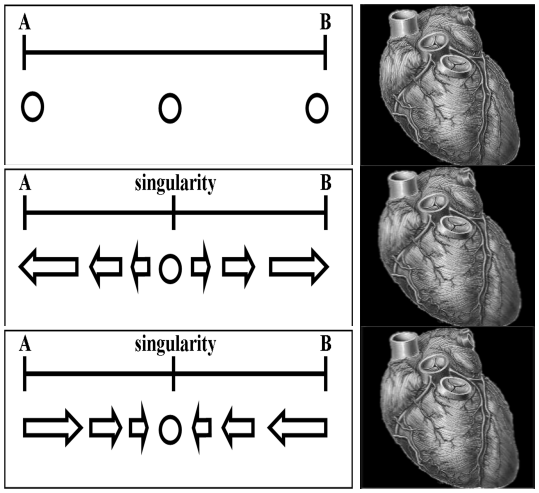


Fig. 12 Pulsating Heart. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1) = (-2, 0.015)$, and $(2, -0.015)$. The singular point, $x_U = 137$, lies at the center of the image.

frame, and $(u_0, u_1, x_U) = (v_0, v_2, y_V) = (3, -0.044, 64)$, for generating the third frame, both from the input image. The images are 128×128. Here we identify a contraction transformation [31]; since, in both cases, $u_1 = v_2$, the singular point is classified as a focus of contraction [20],[21].

Zoom-In: Obtained similarly as Zoom-Out, but with affine parameters $(u_0, u_1, x_U) = (v_0, v_2, y_V) = (-2, 0.031, 64)$ and $(u_0, u_1, x_U) = (v_0, v_2, y_V) = (-4, 0.063, 64)$, respectively, for generating the second frame, and the third (Fig. 14). Here we identify an expansion transformation [31]; since, in both cases, $u_1 = v_2$, the singular point is classified as a focus of expansion [20],[21].

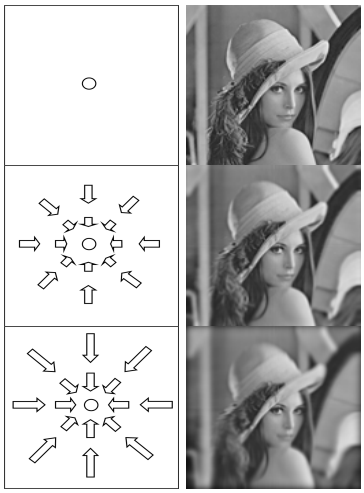


Fig. 13 Zoom-Out. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1) = (v_0, v_2) = (2, -0.03)$ (second frame), and $(u_0, u_1) = (v_0, v_2) = (3, -0.044)$ (third frame), with $x_U = y_V = 64$.

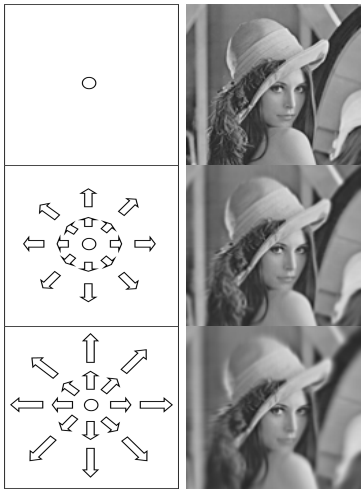


Fig. 14 Zoom-In. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1) = (v_0, v_2) = (-2, 0.031)$ (second frame), and $(u_0, u_1) = (v_0, v_2) = (-4, 0.063)$ (third frame), with $x_U = y_V = 64$.

Deforming Ball: Here (Fig. 15), the same recipe as in Pulsating Heart was followed, except that a larger number of frames was generated. The parameter settings were $(u_0, u_1) = (-16, 0.286)$, $(-12, 0.215)$, $(-8, 0.214)$, $(-4, 0.071)$, $(4, -0.071)$, $(8, -0.214)$, $(12, -0.215)$ and $(16, -0.286)$, respectively. The singular point, $x_U = 56$, lies at the center of the image, which is 113×101.

Funny Eye: This simulation was generated by combining an expansion along the horizontal and a contraction along the vertical. The resulting motion (Fig. 16) is along a hyperbole whose axes are parallel to the image borders and intersect at the image center. The parameter settings were $(u_0, u_1, x_U) = (-2, 0.022, 92)$ and $(v_0, v_2, y_V) = (2, -0.026, 78)$ for the first frame; $(u_0, u_1, x_U) = (-4, 0.043, 92)$ and $(v_0, v_2, y_V) = (4, -0.051, 78)$, for the second, and $(u_0, u_1, x_U) =$

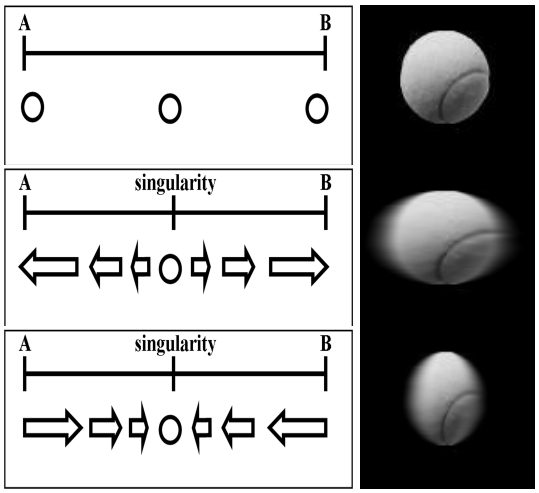


Fig. 15 Deforming Ball. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1) = (-16, 0.286)$, and $(16, -0.286)$, with $x_U = 56$.

$(-6, 0.065, 92)$ and $(v_0, v_2, y_V) = (6, -0.077, 78)$, for the third. The images are 184×156. Here we identify a stretching transformation [31]; since u_1 and v_2 have opposite signs, the singular point is a saddle point [20],[21].

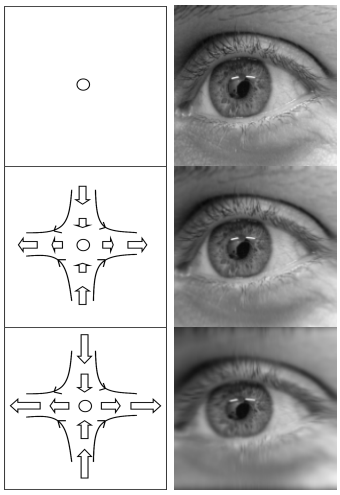


Fig. 16 Funny Eye. From top to bottom: input image and G-filtered sequence for parameter values $(u_0, u_1, x_U) = (-2, 0.022, 92)$ (horizontal) plus $(v_0, v_2, y_V) = (2, -0.026, 78)$ (vertical), for the second frame; and $(u_0, u_1, x_U) = (-6, 0.065, 92)$ (horizontal) plus $(v_0, v_2, y_V) = (6, -0.077, 78)$ (vertical), for the third frame.

5 Conclusions and Future Works

We have reported the study of a data-driven motion synthesis approach based on the one-dimensional affine matching equation, thus extending preliminary work presented in [13,30,32]. Here we have analysed more thoroughly the affine matching operation, recognizing its translation and scale components, and the existence of a singular (fixed) point, whose position plays an important role in motion synthesis. Following that, we considered the Green's function solution to an approximate version of the affine matching equation, and analysed its use for the simulation of a broad class of motions. We have found that the blur entailed by the Green's function operator helps to convey a more compelling motion impression than obtained with the exact matching operator. Moreover, we also found that, by a judicious choice of matching parameters, the Green's function filter allows the simulation of effects such as translation, rotation, zoom-in and zoom-out, and also of complex non-rigid motions, such as those of a squeezing ball or a pulsating heart. Also with relation to the Green's functions, it is worth mentioning that, from a practical point of view, there are advantages in our employing them as a means for solving the approximate matching equations, since the discretization of a differential equation leads to a sparse linear algebraic system with a large condition number, whereas the discretization of an integral operator leads to a dense matrix with a small condition number [33]. Finally, we stress that the image synthesis algorithm, whose application we have reported here, has been greatly improved over its preliminary version presented in [13,30,32].

As compared to the 'motion without movement' of Freeman et al. [5], mentioned in Section 1 as also constituing a fully data-driven approach, we should remark that our algorithm produces motion *with* movement, meaning that we generate real displacement of the image features, and not just an illusion of motion, as conveyed by irradiance modulation.

One of the most promising extensions of the present work lies in the possible use of the Green's functions as a basis for motion representation. We would then be able to project a computed optical flow onto this basis set, in order to derive *motion coefficients* that would indicate the presence and nature of a local motion. The construction of a multi-resolution version of such a representation is also envisioned, what would certainly lead to a reduction of the computational complexity of the process.

In a different vein, we also consider the extension of the present affine approach to two-dimensional matching.

6 Originality and Contributions

The overall contribution of the work reported here is the study of a data-driven motion synthesis approach based on the one-dimensional affine matching equation, thus extending and improving preliminary work presented in [13,30,32]. More specifically, we provide

- A thorough analysis of the affine matching operation, recognizing its translation and scale components, and the existence of a singular (fixed) point, whose position plays an important role in motion synthesis;
- The analysis of a Green's function solution to an approximate version of the affine matching equation, and its use for the simulation of a broad class of effects, such as translation, rotation, zoom-in and zoom-out, and also of complex nonrigid motions, such as those of a squeezing ball or a pulsating heart. Great realism is achieved by such means, especially when simulating fast camera motions, since the Green's function approach allows the introduction of motion blur simultaneously with the motion effect, what cannot be obtained with standard warping techniques;
- A comparison of the Green's function approach with two motion blur simulation techniques also based on the 1D affine model, showing that our results are qualitatively superior.

The work is original not only in its theoretical aspects, pertaining to the analysis of affine matching, but also in what relates to the implementation of the motion synthesis algorithm arising from it, which led to more compelling simulations than achieved before [30].

Acknowledgments

The research reported here has been partially developed at IMPA's VISGRAF Laboratory, with the sponsorship of CAPES, and at the Department of Computer Science at UFBA, with the sponsorship of FAPESB. The first author would like to thank Professor Augusto C. P. L. da Costa, and the secretary Dilson Anunciação, for their support of his work at UFBA. J.R.A. Torreão acknowledges a grant from CNPq-Brasil. The authors would also like to thank Professor Michael Black for allowing the use of his affine optical flow code [34].

References

1. Shinya M, Fournier A (1992) Stochastic Motion – Motion Under the Influence of Wind, *Computer Graphics Forum 11(3)* pp. 119-128.
2. Oziem D, Campbell N, Dalton C, Gibson D, Thomas B (2004) Combining Sampling and Autoregression for Motion Synthesis, *Proc. of the Computer Graphics International Conference*, pp. 510-513.
3. Foster N, Metaxas D (1996) Realistic Animation of Liquids, *CVGIP 58(5)*, pp. 471-483.
4. Freeman W, Adelson E (1991) The Design and Use of Steerable Filters, *IEEE Trans. on PAMI 13(9)*, pp. 891-906.
5. Freeman W, Adelson E, Heeger D (1991) Motion Without Movement, *Computer Graphics 4*, Vol. 25, pp. 27-30.
6. Brostow G J, Essa I (2001) Image-based motion blur for stop motion animation, *Proc. of the* 28^{th} *annual conference on Computer Graphics and Interactive Techniques*, pp. 561-566.
7. Glassner A (1999) An Open and Shut Case Computer Graphics, *IEEE Computer Graphics and Applications* 19, pp. 82-92.
8. Potmesil M, Chakravarty I (1983) Modeling Motion Blur in Computer Generated Images, *Computer Graphics* 17, pp. 389-399.
9. Max N L, Lerner D M (1985) A Two-and-a-Half-D Motion Blur Algorithm, *Computer Graphics* 19, pp. 85-93.
10. Horn B, Schunck B (1981) Determining optical flow, *Artificial Intelligence 17*, pp. 185-203.
11. Lucas D, Kanade T (1981) An Interative Image Registration Technique with an Application to Stereo Vision, *In Proc. Seventh IJCAI, Vancouver*, pp. 674-679.
12. Black M, Anandan P (1996) The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields, *Computer Vision and Image Understanding, CVIU, 63(1)*, pp. 75-104.
13. Torreão J R A (2001) A Green's Function Approach to Shape from Shading, *Pattern Recognition 34*, pp. 2367-2382.
14. Torreão J R A (2003) Geometric-Photometric Approach to Monocular Shape Estimation, *Image and Vision Computing 21*, pp. 1045-1061.
15. Rav-Acha A, Peleg S (2000) Restoration of Multiple Images with Motion Blur in Different Directions, *Workshop on Applications of Computer Vision*, Palm Springs, pp. 22-28
16. Martinsen T, Quintero F M, Skarda D (1996) Plug-in to the GIMP (Open Source Code version 1.22)
17. http://www.gimp.org
18. Zill D G, Cullen M R (1993) Differential Equations with Boundary-Value Problems, *PWS Publishing Company.*
19. Evans L C (1997) Partial Differential Equations, *American Mathematical Society.*
20. Verri A, Girosi F, Torre V (1989) Mathematical Properties of the 2D Motion Field: from Singular Points to Motion Parameters, *Journal of the Optical Society of America A 6(5)*, pp.698-712.
21. Corpetti T, Mémin E, Pérez P (2003) Extraction of Singular Points from Dense Motion Fields: An Analytic Approach, *Journal of Mathematical Imaging and Vision, vol. 19(3)*, pp. 175-198.
22. Ford R, Strickland R (1995) Representing and Visualizing Fluid Flow Images and Velocimetry Data by Nonlinear Dynamical Systems, *CVGIP: Graphical Models and Image Processing 57(6)*, pp. 462-482.
23. Nogawa H, Nakajima Y, Sato Y (1997) Acquisition of Symbolic Description from Flow Fields: A New Approach Based on a Fluid Model, *IEEE Translations on Pattern Analysis and Machine Intelligence 19 (1)*, pp. 58-63
24. Wohn K, Waxman A (1990) The Analytic Structure of Image Flows: Deformation and Segmentation, *Computer Vision and Graphical Image Processing 2 (49)*, pp. 127-151
25. Maurizot M, Bouthemy P, Delyon B, Juditski A, Odobez J (1995) Determination of Singular Points in 2D Deformable Flow Fields, *Proceedings of* 2^{nd} *IEEE International Conference on Image Processing, Vol. 3*, pp. 488-491

26. Foley J, Van Dam A, Feiner S, Hughes J (1990) Computer Graphics: Principles and Practice in C, 2nd Edition, Addison-Wesley systems programming series.
27. Rekleitis I M (1995) Visual Motion Estimation based on Motion Blur Interpretation, *M. Sc. Thesis of Computer Science*, School of Computer Science, McGill University, Montreal.
28. Rekleitis I M (1996) Steerable Filters and Cepstral Analysis for Optical Flow Calculation from a Single Blurred Image, *Vision Interface*, pp. 159-166
29. Rekleitis I M (1996) Optical Flow Recognition from the Power Spectrum of a Single Blurred Image, *Proc of IEEE International Conference on Image Processing*
30. Ferreira Jr P E, Torreão J R A, Carvalho P C P (2004) Data-Based Motion Simulation Through a Green's Function Approach, *Proc. of XVII SIBGRAPI*, pp. 193-199.
31. Jahne B, Hauβecker H, Geiβler P (1999) *Handbook of Computer Vision and Applications*, Vol 2, Academic Press.
32. Ferreira Jr P E, Torreão J R A, Carvalho P C P, Velho L (2005) Video Interpolation Through Green's Functions of Matching Equations, *Proc. of IEEE International Conference on Image Processing.*
33. Beylkin G (1993) Chapter in the book *Wavelets: Mathematics and Applications*, CRC Press.
34. Black M (1996) Area-Based Optical Flow: Robust Affine Regression, *Software available on-line at http://www.cs.brown.edu/people/black/*

Appendix - Numerical Validation of the Experiments

As a means of validating the experiments in Section 4, we have used a software for motion estimation - kindly provided to us by Professor Michael Black - which is based on affine regression [34]. In it, the 2D affine model is expressed as

$$\begin{bmatrix}\tilde{U}\\ \tilde{V}\end{bmatrix} = \begin{bmatrix}\tilde{u_0}\\ -\tilde{v_0}\end{bmatrix} + \begin{bmatrix}\tilde{u_1} & \tilde{u_2}\\ -\tilde{v_1} & \tilde{v_2}\end{bmatrix} \cdot \begin{bmatrix}x-c_x\\ y-c_y\end{bmatrix}, \qquad (23)$$

where $(c_x, c_y)^T$ denotes the coordinates of the central image point. Comparing the above with equation (16), we find the relations

$$\begin{cases} u_0 = \tilde{u_0} - \tilde{u_1}c_x - \tilde{u_2}c_y \equiv u_0^\star \\ u_1 = \tilde{u_1} \\ u_2 = \tilde{u_2} \\ v_0 = -\tilde{v_0} + \tilde{v_1}c_x - \tilde{v_2}c_y \equiv v_0^\star \\ v_1 = -\tilde{v_1} \\ v_2 = \tilde{v_2} \end{cases} \qquad (24)$$

Taking the above into account, we have thus employed Michael Black's program to estimate the affine motion components, in order to compare them with the input parameters of the Green's filter. In each considered sequence, the input image and a synthesized one have been used for this purpose. It should be noted that, since we have restricted ourselves here to a separable 2D affine model, it is expected that we should find $\tilde{u_2} \approx \tilde{v_1} \approx 0$, in all the experiments. Below, we present the validation results only for a subset of the more complex simulated sequences, namely those illustrated in Figs. 12-16.

Zoom-Out: Table 1 presents the optical flow parameters yielded by [34], along with those used as input to the Green's filter. The third frame in Fig.13 has been used. We see that a very good correspondence is obtained in this case, for all the parameters.

Validation Results			
Estimated Parameters		Input Parameters	
$u_0^\star$	3.291884	u_0	3
$\tilde{u_1}$	-0.046732	u_1	-0.044
$\tilde{u_2}$	-0.000591	u_2	0
$v_0^\star$	3.178174	v_0	3
$\tilde{v_1}$	-0.004338	v_1	0
$\tilde{v_2}$	-0.047586	v_2	-0.044

Table 1 Zoom-Out (Fig. 13). On the left-hand side, we present the flow parameters estimated through [34], and, on the right-hand side, the input parameters of the Green's filter. Here $(c_x, c_y) = (64, 64)$.

Zoom-In: Again, the estimated and input parameters are very consistent, as shown by Table 2. The second frame in Fig. 14 has been used.

Validation Results			
Estimated Parameters		Input Parameters	
$u_0^\star$	-2.264054	u_0	-2
$\tilde{u_1}$	0.032386	u_1	0.031
$\tilde{u_2}$	-0.000073	u_2	0
$v_0^\star$	-2.110715	v_0	-2
$\tilde{v_1}$	-0.000666	v_1	0
$\tilde{v_2}$	0.032579	v_2	0.031

Table 2 Zoom-In (Fig. 14). On the left-hand side, we present the flow parameters estimated through [34], and, on the right-hand side, the input parameters of the Green's filter. Here $(c_x, c_y) = (64, 64)$.

Funny Eye: The second frame in Fig. 16 has been used. Table 3 shows the estimated and input parameters. Again, the correspondence is fairly good.

Next, we discuss two examples where the validation through Michael Black's program has not been possible: those of the Pulsating Heart and the Deforming Ball simulations.

Pulsating Heart: Table 4 shows the input parameters and those estimated from the third frame of Fig. 12. The data are inconsistent. A similar situation occurs with the Deforming Ball, as shown below:

Deforming Ball: Table 5 shows the input parameters and those estimated from the second frame of Fig.

Validation Results			
Estimated Parameters		Input Parameters	
$u_0^\star$	-1.628218	u_0	-2
$\tilde{u_1}$	0.016469	u_1	0.022
$\tilde{u_2}$	-0.000277	u_2	0
$v_0^\star$	1.63986	v_0	2
$\tilde{v_1}$	0.000104	v_1	0
$\tilde{v_2}$	-0.020761	v_2	-0.026

Table 3 Funny Eye (Fig. 16). On the left-hand side, we present the flow parameters estimated through [34], and, on the right-hand side, the input parameters of the Green's filter. Here $(c_x, c_y) = (92, 78)$.

Validation Results			
Estimated Parameters		Input Parameters	
$u_0^\star$	16.971032	u_0	2
$\tilde{u_1}$	-0.005220	u_1	-0.015
$\tilde{u_2}$	0.019643	u_2	0
$v_0^\star$	14.340458	v_0	0
$\tilde{v_1}$	0.009142	v_1	0
$\tilde{v_2}$	0.001964	v_2	0

Table 4 Pulsating Heart (Fig. 12). On the left-hand side, we present the flow parameters estimated through [34], and, on the right-hand side, the input parameters of the Green's filter. Here $(c_x, c_y) = (137, 127)$.

15. Again, the data are not consistent, although the errors are somewhat smaller than in the Pulsating Heart experiment.

Validation Results			
Estimated Parameters		Input Parameters	
$u_0^\star$	-12.857846	u_0	-16
$\tilde{u_1}$	0.230788	u_1	0.286
$\tilde{u_2}$	0.039386	u_2	0
$v_0^\star$	0.372565	v_0	0
$\tilde{v_1}$	0.004486	v_1	0
$\tilde{v_2}$	-0.002986	v_2	0

Table 5 Deforming Ball (Fig. 15). On the left-hand side, we present the flow parameters estimated through [34], and, on the right-hand side, the input parameters of the Green's filter. Here $(c_x, c_y) = (56, 51)$.

We conjecture that the problem, in the above simulations, may arise from the fact that, in both cases, the input images consist of the superposition of a central object over a dark background, what could somehow induce errors in the estimation process. In order to check such hypothesis, we performed an additional test based on an image where such a clear-cut figure/background segmentation is not present. For this purpose, we chose the input image to the zoom experiments, applying over it a Green's filter with parameters $(u_0, u_1, x_U) = (2, -0.031, 64)$, in order to simulate only horizontal motion, as in the Deforming Ball and Pulsating Heart examples. The generated pair appears in Fig. 17.

(a) (b)

Fig. 17 Additional Test: (a) Original image. (b) G-filtered image, for parameters $(u_0, u_1, x_U) = (2, -0.031, 64)$.

Table 6, below, shows the estimated and input parameters, which, in this case, prove fairly consistent.

Validation Results			
Estimated Parameters		Input Parameters	
$u_0^\star$	2.248355	u_0	2
$\tilde{u_1}$	-0.032228	u_1	-0.031
$\tilde{u_2}$	-0.000398	u_2	0
$v_0^\star$	0.019306	v_0	0
$\tilde{v_1}$	0.001106	v_1	0
$\tilde{v_2}$	0.000575	v_2	0

Table 6 Additional Test (Fig. 17). On the left-hand side, we present the flow parameters estimated through [34], and, on the right-hand side, the input parameters of the Green's filter. Here $(c_x, c_y) = (64, 64)$.

From the foregoing discussion, we may conclude that, except in the case of images with the characteristics of Figs. 12 and 15 - that is to say, with a sharp figure/background separation -, the Green's function simulations can be numerically validated by the motion estimation algorithm of [34].