Data-Based Motion Simulation Through a Green's Function Approach

Perfilino Eugênio Ferreira Jr.[1], José R. A. Torreão[2] and Paulo Cezar Pinto Carvalho[1]

[1] IMPA–Instituto Nacional de Matemática Pura e Aplicada
Laboratório Visgraf
Estrada Dona Castorina, 110, 22460-320 Rio de Janeiro, RJ, Brasil
perfeuge,pcezar@visgraf.impa.br

[2] Universidade Federal Fluminense
Instituto de Computação
Rua Passo da Pátria, 156, 24210-240 Niterói, RJ, Brasil
jrat@ic.uff.br

Abstract

We describe a data-driven motion simulation algorithm based on an affine image-matching equation. Solving such equation via the Green's function approach, we have obtained a pair of filters which, when applied over an input image, allow the generation of virtual sequences that convey a compelling motion impression. Complex rigid and nonrigid motions have been simulated this way, such as those of a pulsating heart, a twig in the wind and a levitating ball.

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 [1]-[2]. A fully data-driven approach is that of Freeman et al., who use *steerable filters* [3] to create their 'motion without movement' illusion [4]. An alternative, which is exploited here, is to consider solutions of matching (irradiance-conservation) equations of the form

$$I_2(x+u, y+v) = I_1(x,y), \tag{1}$$

where I_1 is the input image, (u,v) is the optical flow field, and I_2 is a matching image, to be found, which, along with I_1, will convey the motion information.

Expanding the left-hand side in a Taylor series up to second order in u and v, and performing a suitable change of variables, it is easy to see that equation (1), for matching along a general direction θ, reduces to the one-dimensional form [5]

$$\frac{u^2}{2} I_2'' + u I_2' + I_2 = I_1, \tag{2}$$

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

Solutions to equation (2), for uniform u, have already been considered in [5], in the context of 3D shape estimation. These can be obtained as the convolution of I_1 with 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 $\delta(x-x_0)$ is substituted for its right-hand side. This has the form

$$G_u(x-x_0) = \frac{2}{u} \sin\left(\frac{x-x_0}{u}\right) e^{-\left(\frac{x-x_0}{u}\right)}, \tag{3}$$

for $x > x_0$, with $G_u = 0$, otherwise. 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 [5].

The Green's function of matching equation (2) has thus already been shown to allow the purely data-driven simulation of a certain kind of 3D movement - namely, that of the irradiance pattern over a static scene, arising from an illumination change. Here we present a way of extending such approach, aiming at the simulation of a broader class of movements. This we do by considering a less restrictive model for our optical flow: instead of the uniform field, $u =$ constant, which led to (3), we introduce the affine model $u + u'x$, where both u and u' are uniform. Considering the solution to the matching equation under such model, we obtain a Green's function filter which, when applied to a given input, is able to generate image sequences that simulate various kinds of uniform and nonuniform motions.

The remainder of this paper is organized as follows: in Section 2, we derive our new Green's function filter, and consider its practical implementation; in Section 3, we present and discuss some experimental results; finally, in Section 4, we make our concluding remarks and propose directions for future work.

2. Affine Matching through Green's Function

Let us go back to the matching equation (2), now considered under an affine optical flow model, *viz.*,

$$\frac{(u+u'x)^2}{2}I_2'' + (u+u'x)I_2' + I_2 = I_1, \tag{4}$$

where it should be kept in mind that both u and u' are assumed as constants, representing, respectively, the optical flow and its spatial rate of change. Similarly as in (2), I_1 and I_2 are taken as functions of $(x, y+\gamma x)$, in order to accomodate matching along a general direction $\theta = \tan^{-1}\gamma$.

Equation (4) has the Cauchy-Euler form [6], and its solution, over a domain D, can be expressed as

$$I_2(x, y+\gamma x) = \int_D G(x,x_0) I_1(x_0, y+\gamma x_0) dx_0, \tag{5}$$

where $G(x,x_0)$ is the Green's function, which solves

$$\frac{(u+u'x)^2}{2}G'' + (u+u'x)G' + G = \delta(x-x_0), \tag{6}$$

given the same boundary conditions as assumed for I_2.

Under the sole condition that $G(x,x_0)$ remains finite as x goes to infinity, a solution to (6) can be found as

$$G(x,x_0) = \frac{2}{u'^2\beta(x_0+a)}\left[\frac{x+a}{x_0+a}\right]^{\alpha} \sin\left\{\beta \log\left[\frac{x+a}{x_0+a}\right]\right\}, \tag{7}$$

for $x > x_0$, with $G(x,x_0) = 0$, otherwise. The parameters a, α and β are given as

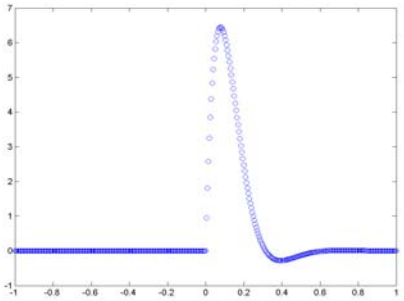
$$\begin{cases} a = \frac{u}{u'} \\ \alpha = -\frac{1}{u'} + \frac{1}{2} \\ \beta = \frac{1}{u'}\sqrt{1 + u' - \frac{u'^2}{4}}, \end{cases} \tag{8}$$

leading to a bounded $G(x,x_0)$ for $D \subset (-a, \infty)$, as long as we take $0 < u' < 2$. In finite domains, this solution is valid for $2 - 2\sqrt{2} < u' < 2 + 2\sqrt{2}$.

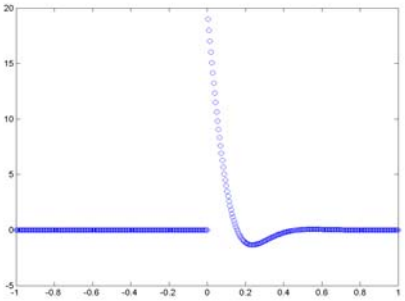
It should be mentioned that, taking a similar form to (7), but with a cosine substituted for the sine term there, we would obtain a second filter - let us call it $H(x,x_0)$ - which, when applied over I_1, leads to a solution of the homogeneous equation associated with (4). Therefore, a general matching pair can be obtained by filtering the input image through a linear combination of G and H, which are the imaginary and real parts, respectively, of the complex function

$$\frac{2}{u'^2\beta(x_0+a)}\left[\frac{x+a}{x_0+a}\right]^{\alpha+i\beta}, \tag{9}$$

for $x > x_0$. Plots of both filters appear in Figure 1.



(a) Filter G with $u = 0.1$ and $u' = 0.002$.



(b) Filter H with $u = 0.1$ and $u' = 0.002$.

Figure 1. Filters were plotted as functions of x, with $x_0 = 0$.

2.1. Implementation Issues

The practical implementation of the Green's function filtering process deserves some discussion. We wish to perform an integration such as (5), and thus, given a certain $\gamma = \tan\theta$, we must traverse the input image, computing weighted sums of the pixel intensities along the direction θ, where the weights will be the G or H values.

There are a couple of ways of achieving this. We could, for instance, use a Bresenham-like incremental line algorithm [7]. Alternatively, we could rotate the image by

an angle $-\theta$, and filter it horizontallly, which is easily done; afterwards, we would revert to the original orientation, through a rotation by θ.

The solution we chose was to visit each image pixel, (x,y), in a raster sequence, and look for the line which crosses that pixel in the required direction. We then performed the summation of products such as $G(x,x_0)I_1(x_0,y_0)$, for all those pixels, (x_0,y_0), lying on that line and preceding the visited site (Fig. 2). The unavoidable aliasing problem, which arises due to the dicretization of the image domain (see Fig. 3), has been dealt with through bilinear and linear interpolation of, respectively, the input image and the Green's function. Algorithm 1, below, describes the whole process (the symbol $*$ stands for the filtering operation defined by (5)). Notice that, instead of trying to locate the actual pixels along the scan line, we uniformly sample points along that line, using a unit sample interval, and compute their intensities using bilinear interpolation.

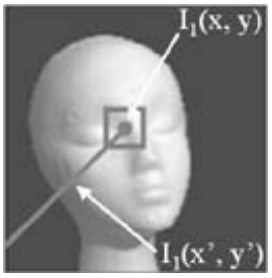


Figure 2. For computing $I_2(x,y)$, we sum the contributions $G(x,x_0)I_1(x_0,y_0)$ from all those sites which precede (x,y) on the scan line of the required direction.

3. Experimental Results

Animated motion sequences illustrating the application of our approach can be found at the web site http://www.impa.br/~perfeuge/green/green.html. Here we briefly discuss some general features of such experiments.

Given a single input image, its companions in each artificial motion sequence have been generated as described in Section 2 above, with u, u', and θ chosen in such a way as to simulate the desired effects. The parameter θ was always picked in the $(-90^\circ, 90^\circ)$ range, that allows us to simulate motions from left to right. Other motions can be created by applying the same algorithms to mirrored or rotated versions of the input image. Motion sequences result more compelling when the values of u and u' are small, since

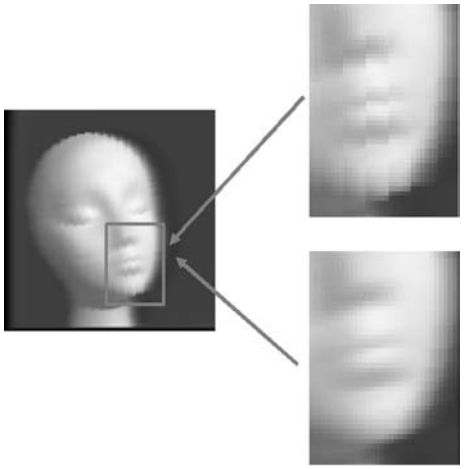


Figure 3. Illustration of aliasing control. The input image was filtered by G, with parameters values $u = 0.03$, $u' = 0.01$ and $\theta = 11^\circ$. The top inset is a zoom of the indicated area, showing the resulting aliasing effect. The bottom one shows its correction through linear interpolation of the Green's function and bilinear interpolation of the input image.

Algorithm 1 Green's function motion simulation

Input: Original Image I_1, Optical Flow u, Spatial Rate of Change u' and the angle θ
Output: $I_2 = I_1 * G$

for each image pixel (x,y) **do**
  $x_{OLD} = x$;
  $y_{OLD} = y$;
  $sum = 0.0$;
  while (x_{OLD}, y_{OLD}) doesn't leave the image **do**
    $x_{OLD} = x_{OLD} - cos(\theta)$;
    $y_{OLD} = y_{OLD} - sin(\theta)$;
    $G(x, x_{OLD}) = LinearInterpolate(u, u')$;
    $pixel = BilinearInterpolate(I_1(x_{OLD}, y_{OLD}))$;
    $sum = sum + G(x, x_{OLD}) \cdot pixel$;
  end while
  $I_2(x,y) = sum$;
end for

Figure 4. Illustration of blur control. The input image, on the top, was filtered by the combination $c_1G + c_2H$, with parameters $u = -0.031$, $u' = -0.001$, and $\theta = 0°$. The middle image resulted for $c_1 = 1.0$ and $c_2 = 0.0$. The image on the bottom resulted for $c_1 = 0.1$ and $c_2 = 5.0$.

the loss of high frequency information is less severe in that case. There is always a certain amount of blur, but it can be controlled by using linear combinations of the G and H filters, as illustrated in Figure 4. This happens because the filtering by H is a homogeneous solution for the affine matching equation. As a consequence, the intensities of pixels at locations x near to x_0 tend to be preserved, since H peaks at that point and the blurring effect of G is attenuated.

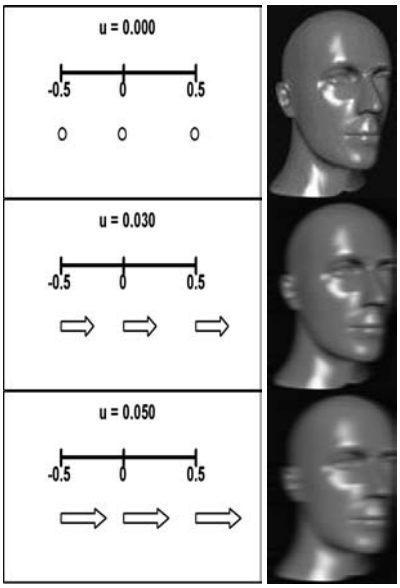
Unless stated otherwise, we will assume that the filtering direction is horizontal ($\theta = 0°$), for simplicity. For the computation of $G(x, x_0)$, we will map the leftmost point in each image row to -0.5 and the rightmost point to 0.5 (thus, the midpoint in each row is mapped to zero).

Next, we will show how to choose the parameters in Equation 4 in order to be able to simulate different classes of movement. We begin by rewriting that equation as

$$\frac{U^2}{2} I_2'' + U I_2' + I_2 = I_1, \tag{10}$$

where $U = u + u'x$, u is the effective disparity or optical flow, and u' is the noneffective disparity or spatial rate of change of u.

The case $u' = 0$ has appeared in [5] and, in that case, Equation 10 reduces to Equation 2. The corresponding Green's function is that given in the Equation 3. As a result of the filterings, one obtains a nonuniform rotation. A translation component can be observed as well. The impression of rotation occurs when an increasing sequence of values of u is given, e.g. $u = 0.01, 0.02, ..., 0.08$. Figure 5 illustrates that behavior. The flow field is uniform, and the object is in the center of the image. Actually, a convincing rotation impression is achieved only if the object is tilted. The blurring effect on the right-side edge of the object boundary plays an important role for this effect.



(a) (b)

Figure 5. (a) Illustration of the behavior of U when $u' = 0$. (b) Motion Simulation. From top to bottom: input image and G_u-filtered sequence for parameter values $u = 0.03$ and 0.05, with $\theta = 0°$.

The general affine matching model (with u' not necessarily equal to zero) allows us to represent much more varied and complex motions. The affine motion model can be expressed by a first order approximation of the optical flow $(U, V)^T$ as

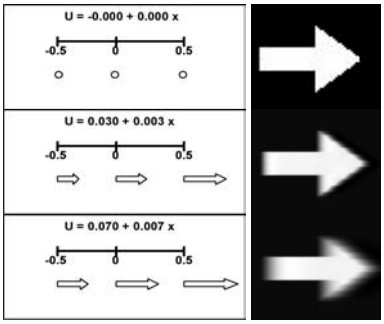
$$\begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} + \begin{bmatrix} D + S_1 & S_2 - R \\ R + S_2 & D - S_1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}, \tag{11}$$

where $(x, y)^T$ are the pixel coordinates in the image, $(u_0, v_0)^T$ are the uniform, zero-order components of the flow, D is a dilation(contraction or expansion) component, R is the rotation component, and S_1 and S_2 are shear components [9]-[10]. As we are concerned with

$U = u + u'x$, it will be assumed that $S_2 - R = 0$, $u = u_0$ and $u' = D + S_1 (u' \neq 0)$. By observing the results, when is kept u constant and $u' \to 0$, the filtering effects of G converge to the effects of G_u. This can also be proved analytically. In this case, there is no dilation and shear components, i.e. $D + S_1 \to 0$. After comparing the sequences created by $u' \neq 0$ and $u' = 0$, we have noted that certain sequences of $u' \neq 0$ convey a better impression of movement. Below we discuss how to generate especific motion simulations, using the $G-$filters.

Translation: can be obtained in two different ways: (a) by a sequence of increasing positive values for u and u'; or (b) by a sequence of decreasing negative values for u and u'; or (c) by sequences where u varies and u' remains constant.

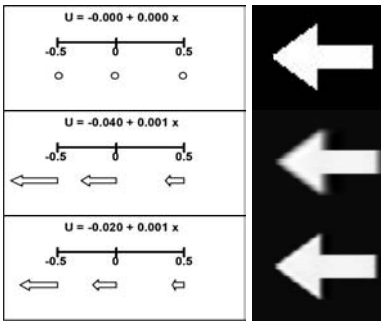
Let us consider the following example of strategy (a): $(u,u') = (0.01,0.001)$, $(0.02,0.002)$, $(0.03,0.003)$, ..., $(0.07,0.007)$. As in the case of the graph in Figure 1(a), one observes that, in each case, G, starting at x_0, first increases, then decreases. As u and u' increase, the rise and fall of G become slower and a larger number of samples x have a nonzero contribution $G(x,x_0)$, causing more blur in the last frames. Figure 6 illustrates the behavior of U in this case. The values U at the rightmost points of the image is larger than those at the leftmost points. Furthermore, as u and u' increase, all points of the image become more displaced towards the right.



(a) (b)

Figure 6. (a) Illustration of the behavior of U. (b) Translation simulation. From top to bottom: input image and G-filtered sequence for parameter values $(u,u') = (0.03,0.003)$, and $(0.07,0.007)$, with $\theta = 0°$.

Strategy (b) is similar to (a). As an example of strategy (c) let us consider: $(u,u') = (-0.040,0.001)$, $(-0.038,0.001)$, $(-0.036,0.001)$, ..., $(-0.008,0.001)$. As u increases, the number of samples x that has a nonzero contribution $G(x,x_0)$ does not change, but the amount of blur decreases. Figure 7 illustrates the behavior of U in this case. As $\|U\|$ always increases, the points of the image are less displaced to right in each succesive frame.

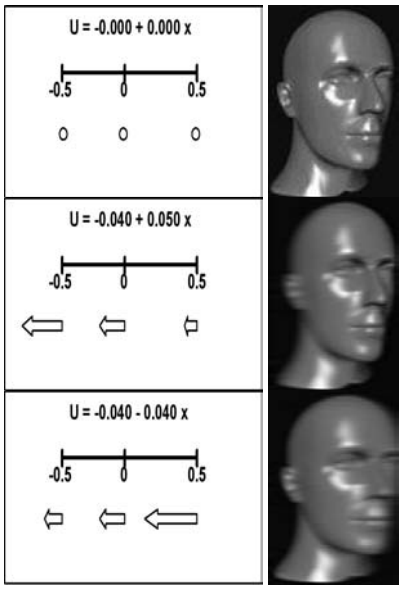


(a) (b)

Figure 7. (a) Illustration of behavior of U. (b) Translation simulation. From up to down: input image and G-filtered sequence for parameter values $(u,u') = (-0.040,0.001)$, and $(-0.020,0.001)$, with $\theta = 0°$.

Rotation: can be obtained by keeping u fixed, while varying u'. Let us consider the following example of this strategy: $(u,u') = (-0.04,-0.04)$, ..., $(-0.04,-0.01)$, $(-0.04,-0.009)$, $(-0.04,-0.005)$, $(-0.04,-0.001)$, $(-0.04,0.001)$, $(-0.04,0.005)$, $(-0.04,0.009)$, $(-0.04,0.01)$, ..., $(-0.04,0.05)$. When u' is negative, $\|U\|$ increases. As a consequence, the rightmost points in each image row are more displaced to right than the leftmost points. This gives an impression of expansion. When u' is positive, $\|U\|$ decreases. As a consequence, the rightmost points in each image row are less displaced to right than the leftmost points, causing an impression of contraction. Figure 8 illustrates the behavior of U in this case.

Pulsating Heart: can be obtained through the following sequence of triples: $(u,u',\theta) = (0.04,0.001,0°)$, $(0.04,0.001,5°)$, and $(-0.02,0.001,0°)$. In this case U is always increasing. Hence the rightmost points in each image row are more displaced to the right than the leftmost points. Notice that U is positive for the two first frames and negative in the last frame. An impression of expansion is achieved with the two first frames, and an impression of contraction is provided by the last frame. In the transition from the second to the third frame two things occur : (1) the values of $\|U\|$ at the rightmost points of the third image are smaller than those at the corresponding points in the second frame; and (2) the values of $\|U\|$ at the leftmost points of the third image are greater than those in the corresponding points in the second frame. This explains the impression of contraction obtained. The perception of movement for the blood vessels is due to the transitions



(a) (b)

Figure 8. (a) Illustration of the behavior of U. (b) Rotation Simulation. From top to bottom: input image and G-filtered sequence for parameter values $(u,u') = (-0.04, 0.05)$, and $(-0.04, -0.04)$, with $\theta = 0°$.

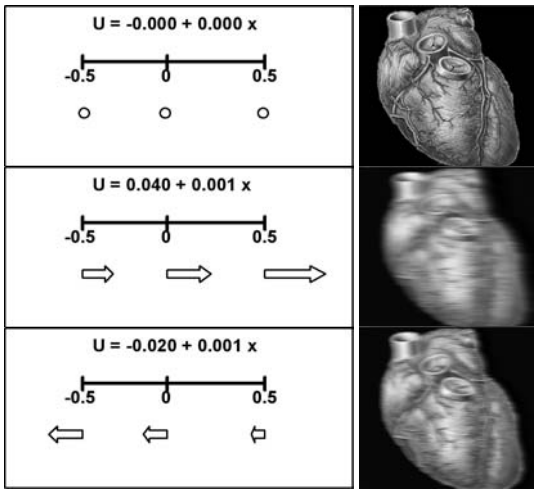
from the first to second frame and from the second to third frame. Second frame only provides a translation motion. Figure 9 illustrates the behavior of U in this case.

Flower under the effect of the wind: can be obtained through the following sequence of triples: $(u,u',\theta) = (0.04, 0.004, 12°)$, $(0.03, 0.003, 9°)$, $(0.02, 0.002, 6°)$, $(0.01, 0.001, 3°)$, $(-0.01, 0.001, -3°)$, and $(-0.02, 0.002, -6°)$. Large values of u and u' provide a perception of movement of wild wind acting on the flower. The idea is that the human eye can not perceive the high frequency informations on the flower under the effect of wild wind. Compelling simulation is achieved because the values of the triple (u,u',θ) decay.

Levitating ball: values of u and u' were fixed at 0.03 and 0.01, respectively. This effect is simpler to obtain, requiring only to vary the θ angle. To generate the example image in our web site we used $\theta \in (-45°, 45°)$.

Earthquake: values of u and u' were fixed at 0.02 and 0.002, respectively. The sequence was generated with nine frames. The first six frames were obtained with $\theta = 0$, 20, -20, 0, 17, and $-17°$. After that, u and u' were fixed at 0.01 and 0.001, respectively, and the θ values set to 0, 14, and $-14°$. The blurring in the first images and their oscilation gives us an impression that an imaginary camcorder was recording the earthquake. The smaller amount of blur in the last images give us an impression of decelerating tremor.

In summary, by a proper combination of parameters, we have been able to achieve plausible simulations of both rigid and nonrigid motions, as illustrated above.



(a) (b)

Figure 9. (a) Illustration of the behavior of U. (b) Pulsating Heart. From top to bottom: input image and G-filtered sequence for parameter values $(u,u') = (0.04, 0.001)$, and $(-0.02, 0.001)$, with $\theta = 0°$.

4. Concluding Remarks

We have presented a data-driven motion simulation approach based on the Green's function solution of an affine matching equation. By filtering an input image through our matching kernels, we have been able to generate virtual image sequences which induce a compelling motion perception. Animations produced this way have been shown to simulate both rigid and nonrigid motions, such as those of a pulsating heart, a levitating ball or a flower in the wind.

As an extension of the present work, a possibility which can be exploited is to apply our Green's function filters with different tuning parameters for different regions of the input image, in order to create cartoon-like animations. The generation of virtual stereoscopic pairs is also envisioned. On a different line, we also intend to test our approach against that of Freeman et al. [4], and to compare our simulations with motions generated by transformations of geometric models.

As a final remark, we should mention that Green's functions of alternative forms of affine matching equations have been considered elsewhere [8], in the context of computer vision 3D shape estimation.

Acknowledgements

We would like to thank the members of Visgraf Laboratory at IMPA, Aristófanes Corrêa, Adelailson Peixoto, and Nair Duarte, for their help with our implementations, and their invaluable comments on previous versions of this work.

References

[1] M. Shinya and A. Fournier, Stochastic Motion – Motion Under the Influence of Wind, *Computer Graphics Forum (Procs. of Eurographics' 92) 11(3)* 119-128 (1992).

[2] N. Foster and D.N. Metaxas, Realistic Animation of Liquids, *CVGIP: Graphical Model and Image Processing 58(5)* 471-483 (1996).

[3] W. Freeman and E. Adelson, The Design and Use of Steerable Filters, *IEEE Transations on Pattern Analysis and Machine Intelligence 13(9)* 891-906 (1991).

[4] W. Freeman, E. Adelson and D. Heeger, Motion Without Movement, *Computer Graphics 4*, Volume 25, 27-30 (1991).

[5] J. R. A. Torreão, A Green's Function Approach to Shape from Shading, *Pattern Recognition 34*, 2367-2382 (2001).

[6] D.G. Zill and M.R. Cullen, *Differential Equations with Boundary-Value Problems*, PWS Publishing Company (1993).

[7] J. Foley, A. Van Dam, S. Feiner and J. Hughes, *Computer Graphics: Principles and Practice in C*, 2nd Edition, Addison-Wesley systems programming series (1990).

[8] J. R. A. Torreão, The Green's Function Approach in Computer Vision, Technical Report RT-11/02, Instituto de Computação, Universidade Federal Fluminense, August 2002. (In portuguese. Available at http://www.ic.uff.br/PosGrad/RelatTec).

[9] C. Capurro, F. Panerai and G. Sandini, Dynamic Vergence Using Log-Polar Images, *International Journal of Computer Vision 24(1)*, 79-94 (1997).

[10] B. Jahne, H. Hauβecker and P. Geiβler, *Handbook of Computer Vision and Applications*, Academic Press (1999).