ISSN:1575-6807
D.L.: GC-1317-1999

INSTITUTO UNIVERSITARIO DE CIENCIAS Y TECNOLOGIAS CIBERNETICAS

IWCVIA'03: International WorkShop on Computer Vision and Image Analysis

Editor: Luis Alvarez

Nº 0026

December 4-6, 2003

Cuadernos del Instituto Universitario de Ciencias y Tecnologías Cibernéticas
Instituto Universitario de Ciencias y Tecnologías Cibernéticas
Universidad de Las Palmas de Gran Canaria
Campus de Tafira
35017 Las Palmas, España
http://www.iuctc.ulpgc.es

IWCVIA'03: International Workshop on Computer Vision and Image Analysis

I

Instituto Universitario de Ciencias y Tecnologías Cibernéticas (IUCYTC)
Universidad de Las Palmas de Gran Canaria (ULPGC)

Las Palmas de Gran Canaria December 4-6, 2003

Organized and Exponsored by the Spanish Research Network: ***RED TEMATICA SOBRE ESTRUCTURACION, PROCESAMIENTO Y GENERACION DE VIDEO DIGITAL EN APLICACIONES MULTIMEDIA*** **(Ref: TIC2001-4570-E)**

General Chairman

- Alvarez Luis

Local Organization Commite in alphabetic order

- Aleman Miguel
- Alvarez Luis
- Cuenca Carmelo
- Esclarín Julio
- González Esther
- Mazorra Luis
- Salgado Agustín
- Sánchez Javier
- Santana Francisco
- Trujillo Agustín

Editorial Commite in alphabetic order

- Alvarez Luis (Universidad de Las Palmas de Gran Canaria)
- Baumela Luis (Universidad Politécnica de Madrid)
- Bouthemy Patrick (INRIA)
- Guil Nicolás (Universidad de Málaga)
- Pascal Fua (Swiss Federal Institute of Technology)
- Perales Francisco (Universitat de les Illes Balears)
- Pérez Nicolás (Universidad de Málaga)
- Pla Filiberto (Universitat Jaume I)
- Weickert Joaquim (Saarland University)

IWCVIA'03 Program.
Hotel AC-Gran Canaria. Salón Gran Forum B

Joaquim Weickert (INVITED SPEAKER), A. Bruhn, N. Papenberg and T. Brox …(Page 1) 04/12/2003 15:15:
Saarland University (Germany)
Variational Optic Flow Computation: From Continuous Models to Algorithms

Nils Papenberg …(Page 7) 04/12/2003 16:30:
Saarland University (Germany)
Numerical Justification for Multiresolution Optic Flow Computation

Jose María González-Linares and Nicolás Guil …(Page 13) 04/12/2003 17:30:
Universidad de Málaga
Contour-based shape retrieval: A 2D Deformable Objects Detection Algorithm

Julián Ramos Cózar, Nicolás Guil and Emilio L. Zapata …(Page 17) 04/12/2003 18:00:
Universidad de Málaga
Affine Transformation Computation of Planar Objects

Luis Alvarez, Luis Mazorra and Francisco Santana …(Page 23) 04/12/2003 18:30:
Universidad de Las Palmas de Gran Canaria
MultiScale Analysis and Shape Representation

Luis Alvarez, Julio Esclarín and Agustín Trujillo …(Page 29) 04/12/2003 19:00:
Universidad de Las Palmas de Gran Canaria
A model based edge location with subpixel precision

Pascal Fua (INVITED SPEAKER) …(Page 33) 05/12/2003 9:30:0
Swiss Federal Institute of Technology, Switzerland
Recovering Human Shape and Motion from Video Sequences

Nicolás Pérez de la Blanca , Jose Manuel Fuertes, Manuel Lucena …(Page 35) 05/12/2003 10:45:
Universidad de Granada and Universidad de Jaen
Tracking 3D moving facial points from calibrated stereo

Jose Maria Buades Rubio, Manuel González Hidalgo, Francisco José Perales López …(Page 43) 05/12/2003 11:45:
Universitat de les Illes Balears
Face and Hands Segmentation in Color Images and Initial Matching

José Miguel Buenaposada, Enrique Muñoz and Luis Baumela …(Page 49) 05/12/2003 12:15:
Universidad Politécnica de Madrid
Real-time robust facial expressions analysis

Enrique Muñoz, Jose Miguel Buenaposada, and Luis Baumela …(Page 55) 05/12/2003 12:45:
Universidad Politécnica de Madrid
Some experiments on incremental image alignment

Patrick Bouthemy (INVITED SPEAKER), G. Piriou, N. Peyrard and J.-F. Yao …(Page 61) 05/12/2003 15:30:
IRISA / INRIA Rennes and IRMAR
Probabilistic models of image motion for recognition of dynamic content in video

Variational Optic Flow Computation: From Continuous Models to Algorithms

Joachim Weickert, Andrés Bruhn, Nils Papenberg, and Thomas Brox
Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science
Saarland University, Building 27
66041 Saarbrücken, Germany
{weickert, bruhn, papenberg, brox}@mia.uni-saarland.de

Abstract

Variational methods belong to the most successful techniques for computing the displacement field in image sequences. In this paper we analyse the different terms in the energy functional and sketch some of our recent contributions in this area.

1. Introduction

Already in 1981, Horn and Schunck introduced the first variational method for computing the displacement field *(optic flow)* in an image sequence [15]. This method is based on two assumptions that are characteristic for many variational optic flow methods: a brightness constancy assumption and a smoothness assumption. These assumptions enter a continuous energy functional whose minimiser yields the desired optic flow field. Performance evaluations such as [5, 11] showed that variational methods belong to the better performing techniques. It is thus not surprising that a lot of research has been carried out in order to improve these techniques even further: These amendments include refined model assumptions with discontiuity-preserving constraints [2, 10, 13, 21, 22, 25, 30] or spatiotemporal regularisation [6, 20, 31], improved data terms with modified constraints [3, 9, 21, 26] or nonquadratic penalisation [6, 14, 18, 29], and efficient multigrid algorithms [7, 12, 27, 32] for minimising these energy functionals.

The goal of the present paper is to analyse the data term and the smoothness term in detail and to survey some recent results on variational optic flow computation in our group. The paper is organised as follows: In Section 2 we sketch the general structure of these techniques. While Section 3 analyses the data term in more detail, a discussion of the different possibilities for smoothness constraints is given in Section 4. Algorithmic aspects are outlined in Section 5, and experiments are presented in Section 6.

2. General Structure

Let $f(x_1, x_2, x_3)$ denote some scalar-valued image sequence, where (x_1, x_2) is the location and x_3 denotes time. Often f is obtained by preprocessing some initial image sequence f_0 by convolving it with a Gaussian K_σ of standard deviation σ:

$$f = K_\sigma * f_0. \tag{1}$$

Let us assume that $D^k f$ describes the set of all partial (spatial and temporal) derivatives of f of order k, and that the optic flow field $u(x_1, x_2, x_3) = (u_1(x_1, x_2, x_3), u_2(x_1, x_2, x_3), 1)$ gives the displacement rate between subsequent frames. In the present paper we consider variational methods that are based on the minimisation of the continuous energy functional

$$E(u) = \int_\Omega (\underbrace{M(D^k f, u)}_{\text{data term}} + \alpha \underbrace{S(\nabla f, \nabla u)}_{\text{regulariser}}) \, dx \tag{2}$$

where the integration domain Ω is either a spatial or a spatiotemporal domain. In the spatial case we have $x := (x_1, x_2)^\top$ and $\nabla := \nabla_2 := (\partial_{x_1}, \partial_{x_2})^\top$, and in the spatiotemporal case we use the notations $x := (x_1, x_2, x_3)^\top$ and $\nabla := \nabla_3 := (\partial_{x_1}, \partial_{x_2}, \partial_{x_3})^\top$. The optic flow field $u(x_1, x_2, x_3)$ is obtained as a function that minimises $E(u)$. The energy functional $E(u)$ penalises all deviations from model assumptions. Typically is consists of a *data term* $M(D^k f, u)$ which expresses e.g. a brightness constancy assumption, and a *regulariser* $S(\nabla f, \nabla u)$ with $\nabla u := (\nabla u_1, \nabla u_2)^\top$ that penalises deviations from (piecewise) smoothness. The weight $\alpha > 0$ serves as *regularisation parameter:* Larger values correspond to more simplified flow fields.

It should be noted that such continuous energy functionals may be formulated in a rotationally invariant way. Results from numerical analysis show that consistent discretisations approximate this invariance under rotations arbitrarily well if the sampling is sufficiently fine. Moreover, if

the energy functional is convex, a unique minimiser exists that can be found in a relatively simple way by globally convergent algorithms. Variational optic flow methods are *global* methods: If there is not sufficient local information, the data term $M(D^k f, u)$ is so small that it is dominated by the smoothness term $\alpha S(\nabla f, \nabla u)$ which fills in information from more reliable surrounding locations. Thus, in contrast to local methods, the *filling-in effect* of global variational approaches always yields dense flow fields and no subsequent interpolation steps are necessary: Everything is automatically accomplished within a single variational framework.

3. Data Terms

Many differential methods for optic flow are based on the assumption that the grey values of image objects in subsequent frames do not change over time. Thus, if $(x_1(x_3), x_2(x_3))$ denotes the movement of some image structure, we obtain the following *optic flow constraint (OFC)* by applying the chain rule:

$$0 = \frac{df(x_1(x_3), x_2(x_3), x_3)}{dx_3} = f_{x_1} u_1 + f_{x_2} u_2 + f_{x_3}, \quad (3)$$

where $f_{x_i} := \partial_{x_i} f$. Note that the optic flow field satisfies $(u_1, u_2, 1)^\top = (\partial_{x_3} x_1, \partial_{x_3} x_2, 1)^\top$. In order to use Equation (3) within the energy functional (2), we penalise all deviations from zero by using the quadratic data term [15]

$$M_1(D^1 f, u) := (u^\top \nabla_3 f)^2. \quad (4)$$

This term can be modified in several ways:

1. One may assume that the optic flow is constant within some neighbourhood of order ρ. This leads to [17]

   $$M_2(D^1 f, u) := K_\rho * ((u^\top \nabla_3 f)^2). \quad (5)$$

   This data term offers advantages when noise is present.

2. Higher robustness under noise can also be achieved by penalising outliers less severely than a quadratic regulariser does: One may use a penaliser $\Psi(s^2)$ that is convex in s and increases less rapidly than quadratic functions, e.g. the regularised TV penaliser [24]

   $$\Psi(s^2) = \sqrt{\varepsilon^2 + s^2}. \quad (6)$$

   This modification transforms M_1 and M_2 into

   $$M_3(D^1 f, u) := \Psi((u^\top \nabla_3 f)^2), \quad (7)$$
   $$M_4(D^1 f, u) := \Psi(K_\rho * ((u^\top \nabla_3 f)^2)). \quad (8)$$

Instead of imposing constancy of the image brightness f along the path $(x_1(x_3), x_2(x_3))$, we may impose constancy of the spatial brightness gradient $(f_{x_1}, f_{x_2})^\top$ along such a path [28]. This gives two equations:

$$u^\top \nabla_3 f_{x_1} = 0, \quad (9)$$
$$u^\top \nabla_3 f_{x_2} = 0. \quad (10)$$

Squaring and adding them produces the data term

$$M_5(D^2 f, u) := \sum_{i=1}^{2} (u^\top \nabla_3 f_{x_i})^2. \quad (11)$$

In a similar way, imposing constancy of the (spatial) Hessian of f gives

$$M_6(D^3 f, u) := \sum_{i=1}^{2} \sum_{j=1}^{2} (u^\top \nabla_3 f_{x_i x_j})^2, \quad (12)$$

and constancy of the (spatial) Laplacian $\Delta_2 f$ yields

$$M_7(D^3 f, u) := (u^\top \nabla_3 (\Delta_2 f))^2. \quad (13)$$

There is no general rule which of these data terms should be preferred. While higher-order derivatives are more sensitive to noise, the data terms M_5, M_6 and M_7 may offer advantages over M_1 when the brightness is not constant. On the other hand, M_1 and M_7 are more appropriate than M_5 and M_6 when non-translatory motion dominates. Thus the choice of the "best" data term will always depend on the specific problem.

4. Smoothness Terms

A taxonomy of the different possibilities to design smoothness constraints has been presented in [30]. It exploits the connection between regularisation methods and diffusion filtering: Minimising the energy functional (2) by means of steepest descend, we obtain a system of diffusion–reaction equations, where the diffusion term results from the regulariser $S(\nabla f, \nabla u)$, and the reaction term is induced by the data term $M(D^k f, u)$:

$$\partial_t u_1 = \partial_{x_1} S_{u_{1,x_1}} + \partial_{x_2} S_{u_{1,x_2}} - \tfrac{1}{\alpha} \partial_{u_1} M, \quad (14)$$
$$\partial_t u_2 = \partial_{x_1} S_{u_{2,x_1}} + \partial_{x_2} S_{u_{2,x_2}} - \tfrac{1}{\alpha} \partial_{u_2} M \quad (15)$$

where $S_{u_{i,x_j}}$ denotes the partial derivative of S with respect to $\partial_{x_j} u_i$. The parameter t in this system of partial differential equations (PDEs) is a pure numerical parameter that should not be confused with the time x_3 of the image sequence. For $t \to \infty$, the steady state of the diffusion–reaction system is given by the Euler–Lagrange equations

$$0 = \partial_{x_1} S_{u_{1x_1}} + \partial_{x_2} S_{u_{1,x_2}} - \tfrac{1}{\alpha} \partial_{u_1} M, \quad (16)$$
$$0 = \partial_{x_1} S_{u_{2x_1}} + \partial_{x_2} S_{u_{2,x_2}} - \tfrac{1}{\alpha} \partial_{u_2} M. \quad (17)$$

Table 1: Taxonomy of optic flow regularisers (see [30]).

Name of Regulariser	$S(\nabla f, \nabla u)$
homogeneous [15]	$\sum_{i=1}^{2} \lvert\nabla u_i\rvert^2$
image-driven, isotropic [2]	$g(\lvert\nabla f\rvert^2) \sum_{i=1}^{2} \lvert\nabla u_i\rvert^2$
image-driven, anisotropic [21]	$\sum_{i=1}^{2} \nabla u_i^\top D(\nabla f)\, \nabla u_i$
flow-driven, isotropic [25]	$\Psi\left(\sum_{i=1}^{2} \lvert\nabla u_i\rvert^2\right)$
flow-driven, anisotropic [30]	$\operatorname{tr} \Psi\left(\sum_{i=1}^{2} \nabla u_i \nabla u_i^\top\right)$

They constitute necessary conditions that a minimiser of $E(u)$ has to satisfy.

Let us now have a closer look at the impact of the regulariser. The simplest regulariser is the *homogeneous* regularisation of Horn and Schunck [15]. This quadratic regulariser of type $S(\nabla u) = |\nabla u_1|^2 + |\nabla u_2|^2$ penalises all deviations from smoothness of the flow field. It can be related to linear diffusion with a constant diffusivity. Thus, the flow field is blurred in a homogeneous way such that motion discontinuities may loose sharpness and get dislocated. It is thus not surprising that people have tried to construct a variety of discontinuity-preserving regularisers. Depending on the structure of the resulting diffusion term, we can classify a regulariser $S(\nabla f, \nabla u)$ as image-driven or flow-driven, and isotropic or anisotropic.

For *image-driven* regularisers, S is not only a function of the flow gradient ∇u but also of the image gradient ∇f. This function is chosen in such a way that it respects discontinuities in the image data. If only the gradient *magnitude* $|\nabla f|$ matters, the method is called *isotropic*. It can avoid smoothing at image edges. An *anisotropic* technique depends also on the *direction* of ∇f. Typically it reduces smoothing across edges of f (i.e. along ∇f), while smoothing along edges of f is still permitted. Image-driven regularisers can be related to linear diffusion processes.

Flow-driven regularisers take into account discontinuities of the unknown flow field u by preventing smoothing at or across flow discontinuities. If the resulting diffusion process uses a scalar-valued diffusivity that only depends on $|\nabla u|^2 := |\nabla u_1|^2 + |\nabla u_2|^2$, it is an *isotropic* process. Cases where also the direction of ∇u_1 and ∇u_2 matters are named *anisotropic*. Flow-driven regularisers lead to nonlinear diffusion processes.

Table 1 gives an overview of the different regularisers. As a rule of thumb, one can expect that flow-driven regularisers offer advantages over image-driven ones for highly textured sequences, where the numerous texture edges create an oversegmentation of the flow field. Moreover, anisotropic methods may give somewhat better results than isotropic ones, since the latter ones are too "lazy" at noisy discontinuities. More details can be found in [30].

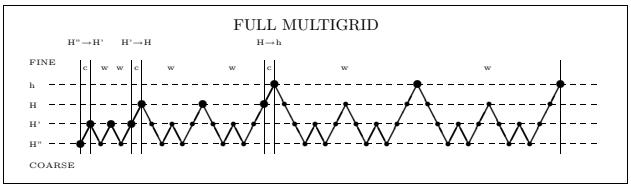


Figure 1: Example of a full multigrid implementation for four levels (from [7]). Starting from a coarse scale the solution is refined step by step.

5. Algorithms

For the numerical minimisation of the energy functional (2), two strategies are used very frequently:

In the first strategy, one discretises the parabolic diffusion–reaction system (14), (15) and recovers the optic flow field as the steady-state solution for $t \to \infty$. The simplest numerical scheme would be an explicit (Euler forward) finite difference scheme. More efficient methods include semi-implicit approaches that offer better stability properties at the expense of the need to solve linear systems of equations.

Alternatively, one can directly discretise the elliptic Euler-Lagrange equations (16), (17). This also requires to solve large linear or nonlinear systems of equations. Efficient methods for this task include *succesive overrelaxation (SOR)* methods, *preconditioned conjugate gradient (PCG)* algorithms and *multigrid* techniques. Figure 1 shows an example of a full multigrid cycle with 4 levels. It has been used in [7, 8] for finding the minimum of a variational approach with data term M_2 and a homogeneous regulariser. On a 3.06 GHz PC, it was possible to compute up to 40 dense flow fields of size 200×200 pixels within a single second. This shows that computational efficieny is no problem for variational optic flow methods, when state-of-the-art numerical methods are used.

It should be noted that for convex energy functionals, there is no danger that any of these two methods gets trapped in a local minimum, since only one minimum exists and the method is globally convergent.

6. Experiments

We start our experiments by evaluating the impact of the data term. This is done in Table 2 where we

Table 2: Impact of the data term on the quality of the optic flow field. We used a spatial energy functional with homogeneous regularisation, and computed the average angular error (AAE) for the Yosemite sequence with clouds. The parameters σ and α have been optimised.

Constancy	Data Term	σ	α	AAE
Brightness	M_1	1.3	500	7.17°
Gradient	M_5	2.1	20	5.91°
Hessian	M_6	2.7	1.8	6.46°
Laplacian	M_7	2.5	3.0	6.18°

used the Yosemite sequence with clouds. This synthetic sequence and its ground truth flow field are available from `ftp://csd.uwo.ca` under the directory `pub/vision`. The experiments in Table 2 show that it can be worthwhile to replace the commonly used brightness constancy constraint by constraints that involve higher derivatives.

The influence of the regulariser is studied in Figure 2, which depicts a zoom into Nagel's *Marble* sequence (`i21www.ira.uka.de/image-sequences`) together with the results for five spatial regularisers. As expected, homogeneous regularisation is fairly blurry, flow-driven regularisers offer advantages over image-driven ones in textured regions, and anisotropic regularisers perform better than isotropic ones.

Figure 3 presents a comparison between spatial and spatiotemporal energy functionals. It demonstrates that the additional assumption of temporal smoothness may lead to significantly improved results.

In Table 3 we juxtapose the angular errors of a number of optic flow methods. It shows that the spatiotemporal method in [29] – which combines the data term M_4 with an isotropic flow-driven regulariser – is one of the two best performing algorithms.

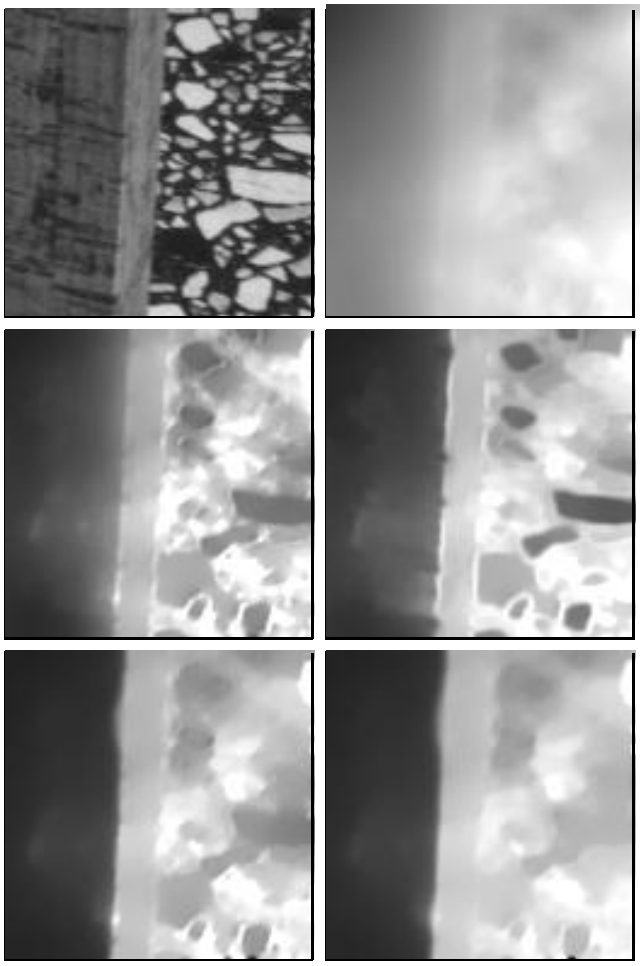


Figure 2: (a) **Top left:** Detail from Frame 16 of the *Marble* sequence (128×128 pixels). (b) **Top right:** Optic flow magnitude for homogeneous regularisation. (c) **Middle left:** Image-driven isotropic regularisation (d) **Middle right:** Image-driven anisotropic regularisation. (e) **Bottom left:** Flow-driven isotropic regularisation (f) **Bottom right:** Flow-driven anisotropic regularisation. From [30].

7. Summary and Extensions

In this paper we have outlined some basic design principles for variational optic flow methods, sketched their numerical implementation and studied their performance. Due to space limitations, we had to restrict ourselves to some of the most important features. There are several possibilities to improve the performance of these methods even further: One may for instance use non-linearised data terms [3, 6, 21], multilevel strategies that encourage convergence towards a global minimiser when nonconvex functionals are applied [3, 4, 18], and consider more sophisticated functionals in order to cope with occlusion problems [1, 23]. On the numerical side, parallelisation strategies can be investigated, e.g. domain decomposition methods [16].

Acknowledgements. Our research has partly been funded by the *Deutsche Forschungsgemeinschaft (DFG)* and the *Graduiertenkolleg "Leistungsgarantien für Rechnersysteme"*. This is gratefully acknowledged.

References

[1] L. Alvarez, R. Deriche, T. Papadopoulo, and J. Sánchez. Symmetrical dense optical flow estimation with occlusion detection. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Computer Vision*

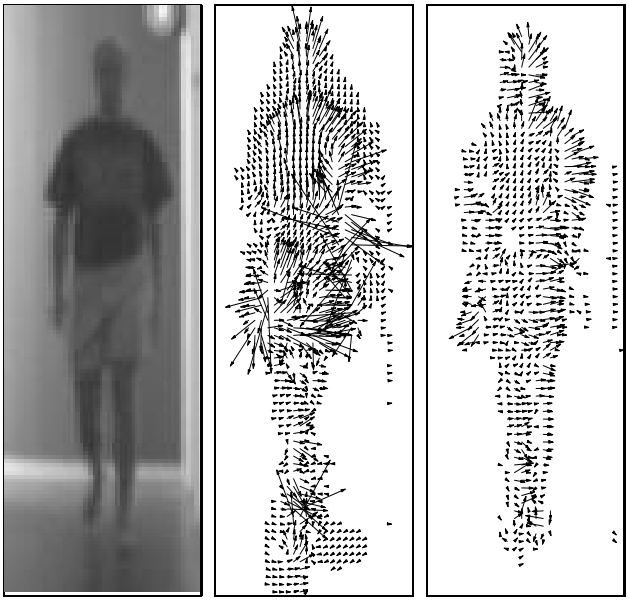


Figure 3: (a) **Left:** Detail of Frame 8 of the *Copenhagen hallway* sequence of size $256 \times 256 \times 16$ pixels. A person is moving towards the camera. (b) **Middle:** Computed flow field for a spatial approach with data term M_1 (brightness constancy) and isotropic flow-driven regularisation. (c) **Right:** Result for a spatiotemporal approach. From [31].

Table 3: Comparison between the results from the literature with 100 % density and our results using a 3-D functional with data term M_4 and an isotropic flow-driven regulariser. All data refer to the *Yosemite* sequence with cloudy sky. Multiscale means that some focusing strategy using linear scale-space or pyramids has been applied. AAE = average angular error. Adapted from [29].

Technique	multiscale	AAE
Horn/Schunck, original [5]	no	31.69°
Singh, step 1 [5]	no	15.28°
Anandan [5]	no	13.36°
Singh, step 2 [5]	no	10.44°
Nagel [5]	no	10.22°
Horn/Schunck, modified [5]	no	9.78°
Uras *et al.*, unthresholded [5]	no	8.94°
Alvarez/Weickert/Sánchez [3]	yes	5.53°
Mémin/Pérez (IEEE TIP) [18]	yes	5.38°
Weickert/Bruhn/Schnörr [29]	no	5.18°
Mémin/Pérez (ICCV '98) [19]	yes	4.69°

– *ECCV 2002*, volume 2350 of *Lecture Notes in Computer Science*, pages 721–736. Springer, Berlin, 2002.

[2] L. Alvarez, J. Esclarín, M. Lefébure, and J. Sánchez. A PDE model for computing the optical flow. In *Proc. XVI Congreso de Ecuaciones Diferenciales y Aplicaciones*, pages 1349–1356, Las Palmas de Gran Canaria, Spain, September 1999.

[3] L. Alvarez, J. Weickert, and J. Sánchez. Reliable estimation of dense optical flow fields with large displacements. *International Journal of Computer Vision*, 39(1):41–56, August 2000.

[4] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[5] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, February 1994.

[6] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proc. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 292–302, Maui, HI, June 1991. IEEE Computer Society Press.

[7] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Real-time optic flow computation with variational methods. In N. Petkov and M. A. Westenberg, editors, *Computer Analysis of Images and Patterns*, volume 2756 of *Lecture Notes in Computer Science*, pages 222–229. Springer, Berlin, 2003.

[8] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Variational optic flow computation in real-time. Technical Report 89, Dept. of Mathematics, Saarland University, Saarbrücken, Germany, June 2003.

[9] A. Bruhn, J. Weickert, and C. Schnörr. Combining the advantages of local and global optic flow methods. In L. Van Gool, editor, *Pattern Recognition*, volume 2449 of *Lecture Notes in Computer Science*, pages 454–462. Springer, Berlin, 2002.

[10] I. Cohen. Nonlinear variational method for optical flow computation. In *Proc. Eighth Scandinavian Conference on Image Analysis*, volume 1, pages 523–530, Tromsø, Norway, May 1993.

[11] B. Galvin, B. McCane, K. Novins, D. Mason, and S. Mills. Recovering motion fields: an analysis of eight optical flow algorithms. In *Proc. 1998 British Machine Vision Conference*, Southampton, England, September 1998.

[12] F. Glazer. Multilevel relaxation in low-level computer vision. In A. Rosenfeld, editor, *Multiresolution Image Processing and Analysis*, pages 312–330. Springer, Berlin, 1984.

[13] F. Heitz and P. Bouthemy. Multimodal estimation of discontinuous optical flow using Markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(12):1217–1232, December 1993.

[14] W. Hinterberger, O. Scherzer, C. Schnörr, and J. Weickert. Analysis of optical flow models in the framework of calculus of variations. *Numerical Functional Analysis and Optimization*, 23(1/2):69–89, May 2002.

[15] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[16] T. Kohlberger, C. Schnörr, A. Bruhn, and J. Weickert. Domain decomposition for parallel variational optic flow computation. In B. Michaelis and G. Krell, editors, *Pattern Recognition*, volume 2781 of *Lecture Notes in Computer Science*, pages 196–203, Berlin, 2003. Springer.

[17] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. Seventh International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, Canada, August 1981.

[18] E. Mémin and P. Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, May 1998.

[19] E. Mémin and P. Pérez. A multigrid approach for hierarchical motion estimation. In *Proc. Sixth International Conference on Computer Vision*, pages 933–938, Bombay, India, January 1998. Narosa Publishing House.

[20] H.-H. Nagel. Extending the 'oriented smoothness constraint' into the temporal domain and the estimation of derivatives of optical flow. In O. Faugeras, editor, *Computer Vision – ECCV '90*, volume 427 of *Lecture Notes in Computer Science*, pages 139–148. Springer, Berlin, 1990.

[21] H.-H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:565–593, 1986.

[22] P. Nesi. Variational approach to optical flow estimation managing discontinuities. *Image and Vision Computing*, 11(7):419–439, September 1993.

[23] M. Proesmans, L. Van Gool, E. Pauwels, and A. Oosterlinck. Determination of optical flow and its discontinuities using non-linear diffusion. In J.-O. Eklundh, editor, *Computer Vision – ECCV '94*, volume 801 of *Lecture Notes in Computer Science*, pages 295–304. Springer, Berlin, 1994.

[24] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[25] C. Schnörr. Segmentation of visual motion by minimizing convex non-quadratic functionals. In *Proc. Twelfth International Conference on Pattern Recognition*, volume A, pages 661–663, Jerusalem, Israel, October 1994. IEEE Computer Society Press.

[26] C. Schnörr. Unique reconstruction of piecewise smooth images by minimizing strictly convex non-quadratic functionals. *Journal of Mathematical Imaging and Vision*, 4:189–198, 1994.

[27] D. Terzopoulos. Image analysis using multigrid relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(2):129–139, March 1986.

[28] S. Uras, F. Girosi, A. Verri, and V. Torre. A computational approach to motion perception. *Biological Cybernetics*, 60:79–87, 1988.

[29] J. Weickert, A. Bruhn, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. Technical Report 82, Dept. of Mathematics, Saarland University, Saarbrücken, Germany, April 2003.

[30] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in PDE-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, December 2001.

[31] J. Weickert and C. Schnörr. Variational optic flow computation with a spatio-temporal smoothness constraint. *Journal of Mathematical Imaging and Vision*, 14(3):245–255, May 2001.

[32] G. Zini, A. Sarti, and C. Lamberti. Application of continuum theory and multi-grid methods to motion evaluation from 3D echocardiography. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 44(2):297–308, March 1997.

Numerical Justification for Multiresolution Optical Flow Computation

Nils Papenberg Andrés Bruhn Thomas Brox
Joachim Weickert
Mathematical Image Analysis Group
Faculty of Mathematics and Computer Science
Saarland University, Building 27, 66041 Saarbrücken, Germany
{papenberg, bruhn, brox, weickert}@mia.uni-saarland.de

Abstract

We introduce a numerical scheme for the minimisation of an energy functional for computing optical flow. This functional combines a brightness constancy assumption, and a discontinuity-preserving spatio-temporal smoothness constraint. In order to allow for large displacements, linearisations in the data term is strictly avoided. The presented numerical scheme is based on two nested fixed point iterations. By proving that this scheme implements a coarse-to-fine warping strategy, we give a theoretical foundation for warping which has been used on a mainly experimental basis so far.

1 Introduction

In the last two decades the quality of optical flow estimation methods has increased dramatically. Starting from the original approaches of Horn and Schunck [11] as well as Lucas and Kanade [15], many new concepts have been developed for dealing with shortcomings of previous models. In order to handle discontinuities in the flow field, the quadratic regulariser in the Horn and Schunck model was replaced by smoothness constraints that permit piecewise smooth results [1, 9, 20, 22, 23]. Some of these ideas are close in spirit to methods for joint motion estimation and motion segmentation [10, 18], and to optical flow methods motivated from robust statistics where outliers are penalised less severely [6, 7]. Spatio-temporal approaches have ameliorated the results by using the information of an additional dimension [19, 6, 24, 10].

Since image sequences are often undersampled in time direction, large displacements are common. In this case nonlinearised models [20, 2] as well as coarse-to-fine strategies [3, 7, 17] have been experimentally demonstrated to be highly useful. Unfortunally – apart from a very nice paper by Lefébure and Cohen [14] – not many results are available that provide a theoretical foundation for this experimentally successful coarse-to-fine warping strategy. The goal of this paper is to close this gap.

To this end we consider a variational method with a nonlinearised data term. The minimizer of this energy functional is approximated by a specific numerical method. This scheme provides a novel foundation for the coarse-to-fine warping that is commonly used in image sequence analysis. This has two important effects: Firstly, it becomes possible to integrate the warping technique, which was so far only algorithmically motivated, into a variational framework. Secondly, it shows a theoretically sound way of how image correspondence problems can be solved with an efficient multi-resolution technique.

The experimental evaluation shows that our method yields excellent results. Compared to those in the literature, their accuracy is always higher.

Paper organisation. In the next section, our variational model is described, first by discussing all model assumptions, and then in form of an energy based formulation. Section 3 derives a minimisation scheme for this energy. The theoretical foundation of warping methods as a numerical approximation step is given in Section 4. An experimental evaluation is presented in Section 5, followed by a brief summary in Section 6.

2 The Variational Model

Before deriving a variational formulation for our optical flow method, we give an intuitive idea of which constraints in our view should be included in such a model.

- **Grey value constancy assumption.**
  Since the beginning of optical flow estimation, it has been assumed that the grey value of a pixel is not changed by the displacement.

$$I(x, y, t) = I(x + u, y + v, t + 1). \tag{1}$$

  Here $I : \Omega \subset \mathbb{R}^3 \to \mathbb{R}$ denotes a rectangular image sequence, and $\mathbf{w} := (u, v, 1)^\top$ is the searched displacement vector between an image at time t and another

image at time $t+1$. The linearised version of the grey value constancy assumption yields the famous optical flow constraint [11]

$$I_x u + I_y v + I_t = 0 \tag{2}$$

where subscripts denote partial derivatives. However, this linearisation is only valid under the assumption that the image changes linearly along the displacement, which is in general not the case, especially for large displacements. Therefore, our model will use the original, non-linearised grey value constancy assumption (1).

- **Smoothness assumption.**
  So far, the model estimates the displacement of a pixel only locally without taking any interaction between neighbouring pixels into account. Therefore, it runs into problems as soon as the gradient vanishes somewhere, or if only the flow in normal direction to the gradient can be estimated (*aperture problem*). Furthermore, one would expect some outliers in the estimates. Hence, it is useful to introduce as a further assumption the smoothness of the flow field. This smoothness constraint can either be applied solely to the spatial domain, if there are only two frames available, or to the spatio-temporal domain, if the displacements in a sequence of images are wanted. As the optimal displacement field will have discontinuities at the boundaries of objects in the scene, it is sensible to generalise the smoothness assumption by demanding a *piecewise smooth* flow field.

- **Multiscale approach.**
  In the case of displacements that are larger than one pixel per frame, the cost functional in a variational formulation must be expected to be multi-modal, i.e. a minimisation algorithm could easily be trapped in a local minimum. In order to find the global minimum, it can be useful to apply multiscale ideas: One starts with solving a coarse, smoothed version of the problem, which may have a unique minimum, hopefully close to the global minimum of the original problem, and uses the result as an initialisation for solving a refined version of the problem. Instead of smoothing the problem, i.e. the image sequence, it is more efficient to downsample the images respecting the sampling theorem, so the model ends up in a multiresolution strategy.

With this description, it is straightforward to derive an energy functional that penalises deviations from these model assumptions. Let $\mathbf{x} := (x, y, t)^\top$ and $\mathbf{w} := (u, v, 1)^\top$. Then the global deviations from the grey value constancy assumption and the gradient constancy assumption are measured by the energy

$$E_{Data}(u, v) = \int_\Omega \left(|I(\mathbf{x} + \mathbf{w}) - I(\mathbf{x})|^2 \right) \mathbf{dx}. \tag{3}$$

Since with quadratic penalisers, outliers get too much influence on the estimation, an increasing concave function $\Psi(s^2)$ is applied, leading to a robust energy [7, 16]:

$$E_{Data}(u, v) = \int_\Omega \Psi\left(|I(\mathbf{x} + \mathbf{w}) - I(\mathbf{x})|^2 \right) \mathbf{dx}. \tag{4}$$

We use the function $\Psi(s^2) = \sqrt{s^2 + \epsilon^2}$ which results in (modified) L^1 minimisation. Due to the small positive constant ϵ, $\Psi(s)$ is still convex which offers advantages in the minimisation process. Moreover, this choice of Ψ does not introduce any additional parameters, since the small numerical parameter ϵ can be set to a fixed value, say 0.001.
Finally, a smoothness term has to describe the model assumption of a piecewise smooth flow field. This is achieved by penalising the total variation of the flow field [21, 8], which can be expressed as

$$E_{Smooth}(u, v) = \int_\Omega \Psi\left(|\nabla_3 u|^2 + |\nabla_3 v|^2 \right) \mathbf{dx}. \tag{5}$$

with the same function for Ψ as above. The spatio-temporal gradient $\nabla_3 := (\partial_x, \partial_y, \partial_t)^\top$ indicates that a spatio-temporal smoothness assumption is involved. For applications with only two images available it is replaced by the spatial gradient.
The total energy is the weighted sum between the data term and the smoothness term

$$E(u, v) = E_{Data} + \alpha E_{Smooth} \tag{6}$$

with some regularisation parameter $\alpha > 0$. Now the goal is to find the functions u and v that minimise this energy.

3 Minimisation

3.1 Euler–Lagrange Equations

Since $E(u, v)$ is highly nonlinear, the minimisation is not trivial. For better readability we define the following abbreviations:

$$\begin{aligned} I_x &:= \partial_x I(\mathbf{x} + \mathbf{w}), \\ I_y &:= \partial_y I(\mathbf{x} + \mathbf{w}), \\ I_z &:= I(\mathbf{x} + \mathbf{w}) - I(\mathbf{x}). \end{aligned} \tag{7}$$

According to the calculus of variations, a minimiser of (6) must fulfill the Euler-Lagrange equations

$$\begin{aligned} 0 &= \Psi'(I_z^2) \cdot (I_x I_z) \\ &\quad -\alpha \operatorname{div} \left(\Psi'(|\nabla_3 u|^2 + |\nabla_3 v|^2) \nabla_3 u \right), \\ 0 &= \Psi'(I_z^2) \cdot (I_y I_z) \\ &\quad -\alpha \operatorname{div} \left(\Psi'(|\nabla_3 u|^2 + |\nabla_3 v|^2) \nabla_3 v \right) \end{aligned}$$

with reflecting boundary conditions.

3.2 Numerical Approximation

The preceding Euler-Lagrange equations are nonlinear in their argument $\mathbf{w} = (u, v, 1)^\top$. A first step towards a linear system of equations, which can be solved with common numerical methods, is the use of fixed point iterations on $\mathbf{w}$. In order to implement a multiscale approach, necessary to better approximate the global optimum of the energy, these fixed point iterations are combined with a downsampling strategy. Instead of the standard downsampling factor of 0.5 on each level, it is proposed here to use an *arbitrary* factor $\eta \in (0, 1)$, what allows smoother transitions from one scale to the next[1]. Moreover, the full pyramid of images is used, starting with the smallest possible image at the coarsest grid. Let $\mathbf{w}^k = (u^k, v^k, 1)^\top$, $k = 0, 1, \ldots$, with the initialisation $\mathbf{w}^0 = (0, 0, 1)^\top$ at the coarsest grid. Further, let I_*^k be the abbreviations defined in (7) but with the iteration variable $\mathbf{w}^k$ instead of $\mathbf{w}$. Then $\mathbf{w}^{k+1}$ will be the solution of

$$\begin{aligned} 0 &= \Psi'((I_z^{k+1})^2) \cdot (I_x^k I_z^{k+1}) \\ &\quad -\alpha \operatorname{div} \left(\Psi'(|\nabla_3 u^{k+1}|^2 + |\nabla_3 v^{k+1}|^2)\nabla_3 u^{k+1}\right) \\ 0 &= \Psi'((I_z^{k+1})^2) \cdot (I_y^k I_z^{k+1}) \\ &\quad -\alpha \operatorname{div} \left(\Psi'(|\nabla_3 u^{k+1}|^2 + |\nabla_3 v^{k+1}|^2)\nabla_3 v^{k+1}\right). \end{aligned} \quad (8)$$

As soon as a fixed point in $\mathbf{w}^k$ is reached, we change to the next finer scale and use this solution as initialisation for the fixed point iteration on this scale.

Notice that we have a fully implicit scheme for the smoothness term and a semi-implicit scheme for the data term. Implicit schemes are used to yield higher stability and faster convergence. However, this new system is still nonlinear because of the nonlinear function Ψ' and the symbols I_*^{k+1}. In order to remove the nonlinearity in I_z^{k+1}, a first order Taylor expansion is used:

$$I_z^{k+1} \approx I_z^k + I_x^k du^k + I_y^k dv^k,$$

where $u^{k+1} = u^k + du^k$ and $v^{k+1} = v^k + dv^k$. So we split the unknowns u^{k+1}, v^{k+1} in the solutions of the previous iteration step u^k, v^k and unknown increments du^k, dv^k. For better readability let

$$\begin{aligned} (\Psi')^k_{Data} &:= \Psi'\left((I_z^k + I_x^k du^k + I_y^k dv^k)^2\right) \\ (\Psi')^k_{Smooth} &:= \Psi'(|\nabla_3(u^k + du^k)|^2 + |\nabla_3(v^k + dv^k)|^2), \end{aligned} \quad (9)$$

where $(\Psi')^k_{Data}$ can be interpreted as a robustness factor in the data term, and $(\Psi')^k_{Smooth}$ as a diffusivity in the smoothness term. With this the first equation in system (8) can be written as

$$\begin{aligned} 0 &= (\Psi')^k_{Data} \cdot \left(I_x^k \left(I_z^k + I_x^k du^k + I_y^k dv^k\right)\right) \\ &\quad - \alpha \operatorname{div} \left((\Psi')^k_{Smooth} \nabla_3(u^k + du^k)\right), \end{aligned} \quad (10)$$

[1]Since the grid size in both x- and y-direction is reduced by η, the image size in fact shrinks with a factor η^2 at each scale.

and the second equation can be expressed in a similar way. This is still a nonlinear system of equations for a fixed k, but now in the unknown increments du^k, dv^k. As the only remaining nonlinearity is due to Ψ', and Ψ has been chosen to be a convex function, the remaining optimisation problem is a convex problem, i.e. there exists a unique minimum solution.

In order to remove the remaining nonlinearity in Ψ', a second, inner, fixed point iteration loop is applied. Let $du^{k,0} := 0$, $dv^{k,0} := 0$ be our initialisation and let $du^{k,l}, dv^{k,l}$ denote the iteration variables at some step l. Furthermore, let $(\Psi')^{k,l}_{Data}$ and $(\Psi')^{k,l}_{Smooth}$ denote the robustness factor and the diffusivity defined in (9) at iteration k, l. Then finally the *linear* system of equations in $du^{k,l+1}, dv^{k,l+1}$ reads

$$\begin{aligned} 0 &= (\Psi')^{k,l}_{Data} \cdot \left(I_x^k \left(I_z^k + I_x^k du^{k,l+1} + I_y^k dv^{k,l+1}\right)\right) \\ &\quad - \alpha \operatorname{div} \left((\Psi')^{k,l}_{Smooth} \nabla_3(u^k + du^{k,l+1})\right) \end{aligned} \quad (11)$$

for the first equation. Using standard discretisations for the derivatives, the resulting sparse linear system of equations can now be solved with common numerical methods, such as Gauss-Seidel or SOR iterations. Expressions of type $I(\mathbf{x} + \mathbf{w}^\mathbf{k})$ are computed by means of bilinear interpolation.

4 Relation to Warping Methods

Coarse-to-fine warping techniques are a frequently used tool for improving the performance of optic flow methods [3, 7, 18]. While they are often introduced on a purely experimental basis, we show in this section that they can be theoretically justified as a numerical approximation. Let us simplify the model by assuming solely spatial smoothness, as in [18]. Under these condition, (10) can be written as

$$\begin{aligned} -(\Psi')^k_{Data} I_z^k \nabla I^k &= (\Psi')^k_{Data} \nabla I^k (\nabla I^k)^\top \begin{pmatrix} du^k \\ dv^k \end{pmatrix} \\ &\quad -\alpha \begin{pmatrix} \operatorname{div}\left((\Psi')^k_{Smooth} \nabla(u^k + du^k)\right) \\ \operatorname{div}\left((\Psi')^k_{Smooth} \nabla(v^k + dv^k)\right) \end{pmatrix} \end{aligned}$$

For a fixed k, this system is equivalent to the Euler–Lagrange equations described in [18]. Also there, only the increments du and dv between the first image and the warped second image are estimated. The same increments appear in the outer fixed point iterations of our approach in order to resolve the nonlinearity of the grey value constancy assumption. *This shows that the warping technique implements the minimisation of a non-linearised constancy assumption by means of fixed point iterations on* $\mathbf{w}$.

In earlier approaches, the main motivation for warping has been the coarse-to-fine strategy. Due to solutions u and v computed on coarser grids, only an increment du and

dv had to be computed on the fine grid. Thus, the estimates used to have a magnitude of less than one pixel per frame, independent of the magnitude of the total displacement. This ability to deal with larger displacements proved to be a very important aspect in differential optical flow estimation.

A second strategy to deal with large displacements has been the usage of the non-linearised grey value constancy assumption [20, 2]. Here, large displacements are allowed from the beginning. However, the nonlinearity results in a multi-modal functional. In such a setting, the coarse-to-fine strategy is not only wanted, but even necessary to better approximate the global minimum. At the end, both strategies not only lead to similar results. In fact, as we have seen above, they are completely equivalent. As a consequence, the coarse-to-fine warping technique can be formulated as a single minimisation problem, and image registration techniques relying on non-linearised constancy assumptions get access to an efficient multiresolution method for minimising their energy functionals.

5 Evaluation

For evaluation purposes, experiments with both synthetic and real-world image data were performed. The presented angular errors were computed via

$$\arccos\left(\frac{u_c u_e + v_c v_e + 1}{\sqrt{(u_c^2 + v_c^2 + 1)(u_e^2 + v_e^2 + 1)}}\right) \qquad (12)$$

where the subscripts c and e denote the correct resp. the estimated flow (cf. [5]).

Let us start our evaluation with the two variants of a famous sequence: the *Yosemite* sequence with and without cloudy sky. The original version of the sequence with cloudy sky was created by *Lynn Quam* and is available at `ftp://csd.uwo.ca` under the directory `pub/vision`. It depicts a flight through the Yosemite national park and combines divergent and translational motion. The version without clouds is available at `http://www.cs.brown.edu/people/black/images.html`.

Tab.1 shows a comparison of our results for both sequences to the best results from the literature. As one can see, our variational approach outperforms all other methods. The corresponding flow fields presented in Fig.1 give a qualitative impression of these raw numbers: They match the ground truth well.

In a second experiment we compare the results of our new linearisation-method to those which are based on the linearised modell using the optical flow constraint (2). Both modells include the same smoothness assumption so differences in the result are only ascribed to the later linerisation of the data term. The outcome listed in Table 2 shows that our methods leads to results which are about 35% better than those from the linearised case. So shifting the linearisation to the numerical approximation improves the accurancy of the algorithm even if there are no large displacements as in the Yosemite sequence.

Table 1: Comparison between the best results from the literature with 100 % density and our results for the *Yosemite* sequence with and without cloudy sky. AAE = average angular error. STD = standard deviation. 2D = spatial smoothness assumption. 3D = spatio-temporal smoothness assumption.

Yosemite with clouds

Technique	AAE	STD
Nagel [5]	10.22°	16.51°
Horn–Schunck, mod. [5]	9.78°	16.19°
Uras *et al.* [5]	8.94°	15.61°
Alvarez *et al.* [2]	5.53°	7.40°
Mémin–Pérez [17]	4.69°	6.89°
Our method (2D)	**4.63**°	**6.89**°
Our method (3D)	**3.94**°	**6.28**°

Yosemite without clouds

Technique	AAE	STD
Ju *et al.* [12]	2.16°	2.00°
Bab-Hadiashar–Suter [4]	2.05°	2.92°
Lai–Vemuri [13]	1.99°	1.41°
Our method (2D)	**1.72**°	**1.37**°
Mémin–Pérez [17]	1.58°	1.21°
Farnebäck [10]	1.14°	2.14°
Our method (3D)	**1.09**°	**1.13**°

For evaluating the performance of our method for real-world image data, the *Ettlinger Tor* traffic sequence by Nagel was used. This sequence consists of 50 frames of size 512×512. It is available at `http://i21www.ira.uka.de/image_sequences/`. In Fig. 2 the computed flow field and its magnitude are shown. Our estimation gives very realistic results, and the algorithm hardly suffers from interlacing artifacts that are present in all frames. Moreover, the flow boundaries are rather sharp and can be used directly for segmentation purposes by applying a simple thresholding step.

6 Conclusion

In this paper we have present a new numerical scheme for the minimisation of a continuous, rotationally invariant energy functional for optical flow computation based on two terms: a robust data term with a brightness constancy, com-

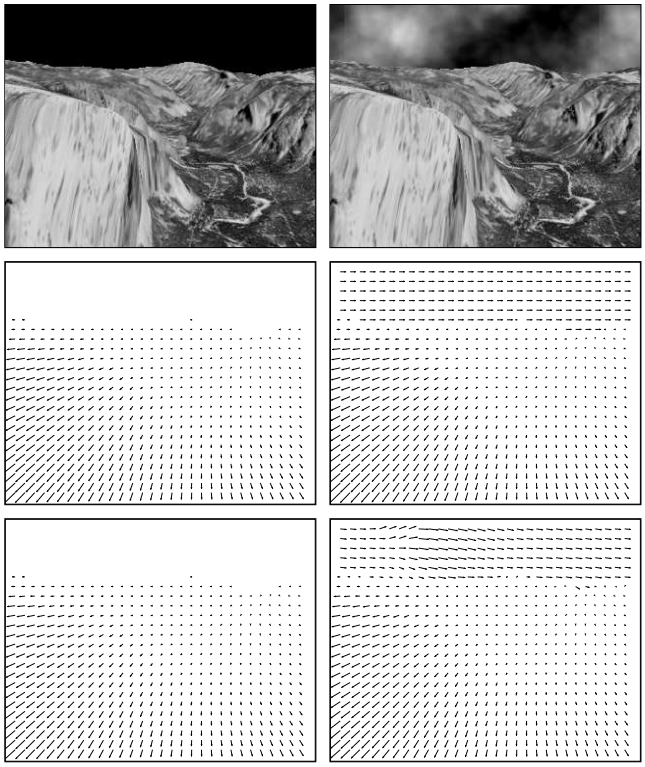


Figure 1: *(a) Top left:* Frame 8 of the *Yosemite* sequence without clouds. *(b) Top right:* Corresponding frame of the sequence *with* clouds. *(c) Middle left:* Ground truth without clouds. *(d) Middle right:* Ground truth *with* clouds. *(e) Bottom left:* Computed flow field by our 3D method for the sequence without clouds. *(f) Bottom right:* Ditto for the sequence *with* clouds.

bined with a discontinuity-preserving spatio-temporal TV regulariser. It should be stressed that we have avoided any linearisations in the data term in order to allow also for large displacements. We have shown that their combination outperforms all methods from the literature so far. One of the main reasons for this performance is the use of an energy functional with *non-linearised* data term and our strategy to consequently postpone all linearisations to the *numerical* scheme: *While linearisations in the model immediately compromise the overall performance of the system, linearisations in the numerical scheme can help to improve convergence to the global minimum.* The important result in our paper is the proof that *the widely-used warping can be theoretically justified as a numerical approximation strategy that does not influence the continuous model*.
As further work we want to transfer this numerical strategy to non-linearised data terms which are based on derivatives of the image sequence function. Our goal is to enhance the estimation of the flow field in areas where the grey value constancy assumption is disturbed like the area of the clouds in the Yosemite sequence.

Table 2: Comparison between our numerical linearisation-method and algorithms using the optical flow constraint as a linearisation of the model. AAE = average anngular error. STD = standard deviation. 2D = spatial smoothness assumption. 3D = spatio-temporal smoothness assumption.

Yosemite with clouds

	our method		ofc-method	
Technique	AAE	STD	AAE	STD
2D	4.63°	6.89°	6.12°	8.49°
3D	3.94°	6.28°	5.43°	8.24°

Yosemite without clouds

	our method		ofc-method	
Technique	AAE	STD	AAE	STD
2D	1.72°	1.37°	2.40°	1.94°
3D	1.09°	1.13°	1.57°	1.48°

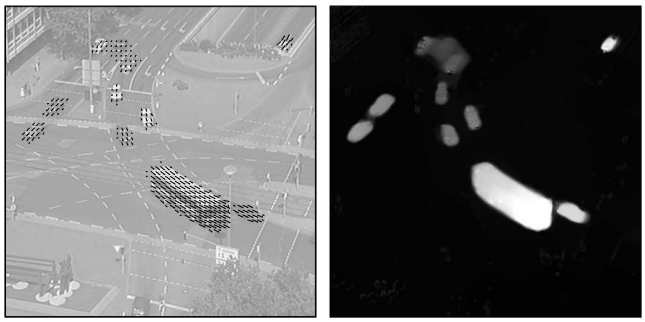


Figure 2: *(a) Left:* Computed flow field between frame 5 and 6 of the *Ettlinger Tor* traffic sequence. *(b) Right:* Computed magnitude of the optical flow field.

Acknowledgements

Our optical flow research is partly funded by the Graduiertenkolleg *Leistungsgarantien für Computersysteme* and the *Deutsche Forschungsgemeinschaft (DFG)*. This is gratefully acknowledged.

References

[1] L. Alvarez, J. Esclarín, M. Lefébure, and J. Sánchez. A PDE model for computing the optical flow. In *Proc. XVI Congreso de Ecuaciones Diferenciales y Aplicaciones*, pages 1349–1356, Las Palmas de Gran Canaria, Spain, Sept. 1999.

[2] L. Alvarez, J. Weickert, and J. Sánchez. Reliable estimation of dense optical flow fields with large displacements. *International Journal of Computer Vision*, 39(1):41–56, Aug. 2000.

[3] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[4] A. Bab-Hadiashar and D. Suter. Robust optic flow computation. *International Journal of Computer Vision*, 29(1):59–77, Aug. 1998.

[5] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, Feb. 1994.

[6] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proc. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 292–302, Maui, HI, June 1991. IEEE Computer Society Press.

[7] M. J. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, Jan. 1996.

[8] I. Cohen. Nonlinear variational method for optical flow computation. In *Proc. Eighth Scandinavian Conference on Image Analysis*, volume 1, pages 523–530, Tromsø, Norway, May 1993.

[9] R. Deriche, P. Kornprobst, and G. Aubert. Optical-flow estimation while preserving its discontinuities: a variational approach. In *Proc. Second Asian Conference on Computer Vision*, volume 2, pages 290–295, Singapore, Dec. 1995.

[10] G. Farnebäck. Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field. In *Proc. Eighth International Conference on Computer Vision*, volume 1, pages 171–177, Vancouver, Canada, July 2001. IEEE Computer Society Press.

[11] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[12] S. Ju, M. Black, and A. Jepson. Skin and bones: multilayer, locally affine, optical flow and regularization with transparency. In *Proc. 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 307–314, San Francisco, CA, June 1996. IEEE Computer Society Press.

[13] S.-H. Lai and B. C. Vemuri. Reliable and efficient computation of optical flow. *International Journal of Computer Vision*, 29(2):87–105, Oct. 1998.

[14] M. Lefébure and L. D. Cohen. Image registration, optical flow and local rigidity. *Journal of Mathematical Imaging and Vision*, 14(2):131–147, Mar. 2001.

[15] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. Seventh International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, Canada, Aug. 1981.

[16] E. Mémin and P. Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, May 1998.

[17] E. Mémin and P. Pérez. A multigrid approach for hierarchical motion estimation. In *Proc. Sixth International Conference on Computer Vision*, pages 933–938, Bombay, India, Jan. 1998. Narosa Publishing House.

[18] E. Mémin and P. Pérez. Hierarchical estimation and segmentation of dense motion fields. *International Journal of Computer Vision*, 46(2):129–155, 2002.

[19] H.-H. Nagel. Extending the 'oriented smoothness constraint' into the temporal domain and the estimation of derivatives of optical flow. In O. Faugeras, editor, *Computer Vision – ECCV '90*, volume 427 of *Lecture Notes in Computer Science*, pages 139–148. Springer, Berlin, 1990.

[20] H.-H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:565–593, 1986.

[21] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[22] C. Schnörr. Segmentation of visual motion by minimizing convex non-quadratic functionals. In *Proc. Twelfth International Conference on Pattern Recognition*, volume A, pages 661–663, Jerusalem, Israel, Oct. 1994. IEEE Computer Society Press.

[23] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in PDE-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, Dec. 2001.

[24] J. Weickert and C. Schnörr. Variational optic flow computation with a spatio-temporal smoothness constraint. *Journal of Mathematical Imaging and Vision*, 14(3):245–255, May 2001.

Contour-based shape retrieval: A 2D Deformable Objects Detection Algorithm

J.M. González-Linares N. Guil

Department of Computer Architecture
Complejo Politécnico, Campus de Teatinos
Apdo. 4114, E-29080 Málaga,Spain

Abstract

This paper presents an algorithm for the automatic detection of 2D objects in an image even in the presence of noise, occlusion, cluttering and/or deformations. This method is based on shape information extracted from the edges gradient and only needs a template of the object to be located. This shape information is invariant to rotation, scale and displacement. This algorithm can be used in contour-based shape retrieval applications.

1. Introduction

Some applications like image database retrieval need automatic tools capable of detecting objects included in complex scenes where the presence of other objects, noise or the deformable nature of the object hinder the process. Typically, the only available information is its 2D shape and few physical features can be assumed.

There are different shape descriptors available, like the ones defined in the MPEG-7 standard. In this standard, three descriptors are proposed [1], one based in region information, another based in contour information, and a third one based in 3-D information. The contour-based shape descriptor uses the Curvature Scale-Space (CSS) representation of the contour [9]. This descriptor have very good features, like its robustness to deformation or that it emulates well the shape similarity perception of the human visual systems. It has some drawbacks too, like its poor robustness to occlusion and cluttering or that it needs closed-contours.

In this work it is presented an algorithm for the automatic detection of 2D objects in an image even in the presence of noise, occlusion, cluttering and/or deformations. The object can be curved or polygonal, closed or open, and simply-connected or multiply-connected. The rest of the paper is organized as follows. In the next section a brief overview of the algorithm is presented. In section 3 some examples are shown and, finally, in section 4 a discussion and an outline of future work is given.

2. Object detection

The algorithm presented in this work tries to detect an object in an image using its shape information. In [8], shape is defined as

Definition 1 Shape *is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object.*

These effects are the Euclidean similarity transformations of displacement, scaling and rotation.

Object detection can be accomplished by using some feature invariant to these transformations. The similarity transformations are invariant to lengths, angles and areas, [10]. In [4], a particular angle called *spatial angle* is proposed to obtain a signature of the image that is invariant to the similarity transformations.

Definition 2 *Let α_{ij} be the* **spatial angle** *between the straight line that joins two edge points, e_i and e_j, and the gradient angle of e_i,*

$$\alpha_{ij} = \arctan \frac{y_i - y_j}{x_i - x_j} \angle \theta_i \tag{1}$$

where (x_i, y_i) and (x_j, y_j) are the coordinates of e_i and e_j, θ_i is the angle of the gradient vector of e_i, and $a \angle b$ means the positive angle between a and b.

The edge points are the local maxima of the gradient in the edge normal direction ([3]). In this work, an edge point, e_i, is characterized by its spatial coordinates, x_i and y_i, and the angle of its gradient vector, $\theta_i = \Phi(x_i, y_i)$. The unordered set of edge points in an image is $\mathbf{E} = \{e_i | \, i \in [1, N]\}$ where N is the number of edge points. These edge points form the contours of every object in the image, and there is no geometric restriction on these objects (they can be curved or polygonal, open or closed, simply-connected or multiply-connected).

It can be shown that the spatial angle is invariant to the similarity transformations of displacement, scale and rotation.

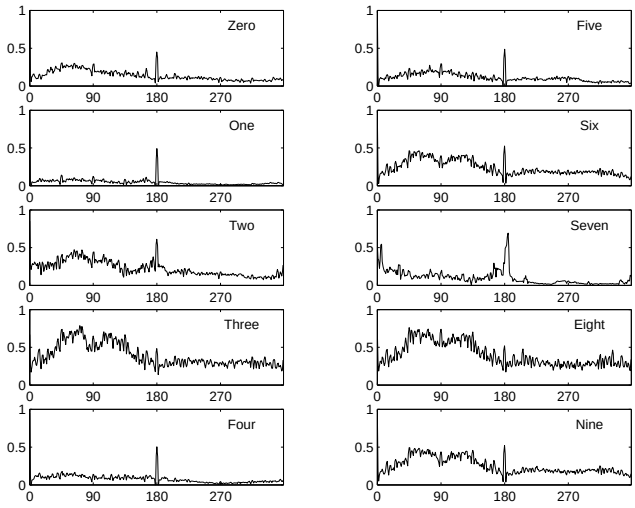


Figure 1: Spatial angles histogram of digits 0 to 9

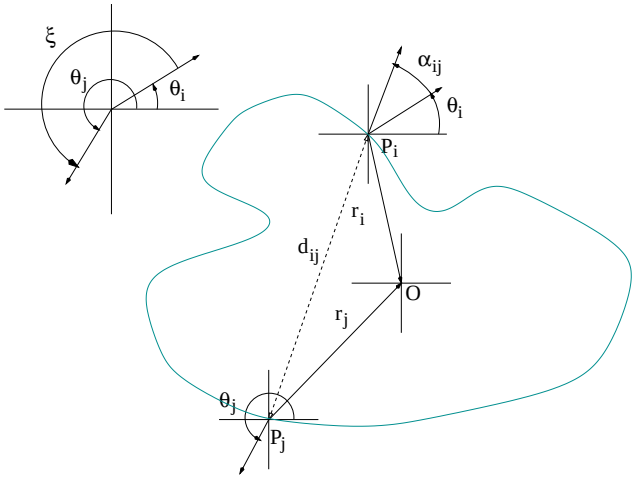


Figure 2: Geometrical description of the θ_i, α_{ij}, and ξ angles

2.1 Image signature

The spatial angle can be used to obtain a signature of a shape. If a set $\mathbf{P}$ is constructed, with every pair of edge points in $\mathbf{E}$,

$$\mathbf{P} = \{(e_i, e_j) |\, i \neq j,\, e_i, e_j \in \mathbf{E}\}$$

then the spatial angle can be computed for every pair and an histogram of these spatial angles ($\mathcal{SAH}$, Spatial Angles Histogram) can be used as the signature of the shape. In Figure 1 the $\mathcal{SAH}$ for several images are shown. The images are the digits 0 to 9. For example, the histograms for digits 6 and 9 are near identical as one digit is a rotated version of the other, but the histograms for digits 6 and 7 are significantly different.

Unfortunately, when there is a large number of edge points, it can be unpractical to obtain this histogram and very difficult to differentiate different objects in it.

A more shape-representative information can be obtained if a subset of spatial angles is selected. This subset must have enough information to discriminate the shape of the object. The description in Figure 2 can be used to define a *difference angle*, ξ, equal to the positive difference between the gradient angles of two edge points. Given a set of fixed values for the difference angle, Ψ, it can be obtained a set $\mathbf{P}_\Psi$ with every pair of edge points such that

$$\mathbf{P}_\Psi = \{(e_i, e_j) |\, i \neq j,\, e_i, e_j \in \mathbf{E},\, \theta_j - \theta_i = \xi,\, \xi \in \Psi\} \quad (2)$$

The difference angles must be carefully selected to obtain a good signature of the object ([5]). In Figure 3.a are shown the histograms ($\mathcal{SAH}_\Psi$) for digits 0 to 9 using only the edge pairs in $\mathbf{P}_\Psi$ with $\Psi = \{90, 180\}$. The histograms of different digits are still different but the computational cost for obtaining the histograms is lesser. A comparison using a method like the earth mover distance [11] between the signatures of two shapes can be done to detect the presence of the object of interest in the other image. In this work a correlation function has been developed to compare these histograms.

Sampling errors, noise or small deformations can spread the spatial angles around its ideal values. To take this into account, a small window will be used. The proposed normalized cross-correlation function can be defined as

$$C = \frac{\sum_\alpha \mathcal{SAH}_\Psi(I) W[\mathcal{SAH}_\Psi(T)]}{NI * NT} \quad (3)$$

$$NI = \sqrt{\sum_\alpha \mathcal{SAH}_\Psi(I) W[\mathcal{SAH}_\Psi(I)]}$$
$$NT = \sqrt{\sum_\alpha \mathcal{SAH}_\Psi(T) W[\mathcal{SAH}_\Psi(T)]}$$
$$W[F(i)] = \sum_w F(i+w)$$

where $\mathcal{SAH}_\Psi(I)$ is the $\mathcal{SAH}_\Psi$ for the image, and $\mathcal{SAH}_\Psi(T)$ for the template. The window function performs an unidimensional summation in $[-W, W]$. The spatial angle is periodic, thus the window function must take it into account.

3. Examples

The correlation values obtained for the digits example are shown in Figure 3.b. Obviously, the diagonal values are 1 and the table is symmetric. The maximum correlation is found between 6 and 9 and the minimum between 3 and 7.

As another example, a database composed by 1101 trademark images has been selected [7]. A similar example was used in [1] to test the performance of region-based shape descriptors. A new trademark, similar to some of the

existing trademarks, has been drawn and compared with the set. The images in the database have been ordered using the correlation value C obtained, and the best 8 matches and the worst 4 are shown in Figure 4. The leftmost image is the new trademark, and the most similar trademarks have a correlation value over 0.95. The comparison process takes between 1-2 s for the entire database.

4. Summary and Conclusions

The shape descriptor proposed in this work can be used to detect objects that suffer similarity transformations even in the presence of noise, occlusion, cluttering and/or deformations. It uses gradient information and has no restriction about the object shape.

In the future we are planning to improve the method in two directions. On the one hand, we want to incorporate local information to improve the shape similarity perception. On the other hand, we want to reduce the amount of information that needs to be stored for every shape.

References

[1] Introduction to MPEG-7. Edited by B.S. Manjunath, Philippe Salembier, and Thomas Sikora. *John Wiley & Sons*, 2002

[2] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[3] John Canny. Finding edges on lines. Technical Report 20, Massachusetts Institute of Technology, 1983.

[4] J.M. Gonzlez-Linares, N. Guil, and E.L. Zapata. An efficient 2D deformable objects detection and location algorithm. *Pattern Recognition*, 36:2543–2556, 2003.

[5] N. Guil, J.M. Gonzlez-Linares, and E.L. Zapata. Bidimensional shape detection using an invariant approach. *Pattern Recognition*, 32:1025–1038, 1999.

[6] Anil K. Jain, Yu Zhong, and S. Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):267–277, March 1996.

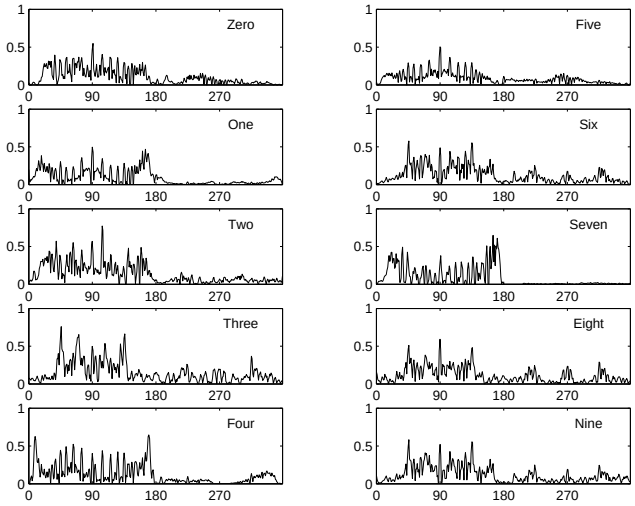
[7] Anil K. Jain, and Aditya Vailaya. Shape-based retrieval: a case study with trademarks image databases. *Signal Processing*, 71(2):109–129, 1998.

[8] D. G. Kendall. The diffusion of shape. *Advances in Applied Probability*, 9:428–430, 1977.

[9] F. Mokhtarian, and A.K. Mackworth A theory of multiscale, curvature-based shape representation for planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):789–805, 1992.

[10] J.L. Mundy and A. Zisserman. *Geometric Invariance in Computer Vision*. MIT Press, 1992.

[11] Rubner, Tomasi, and Guibas. The earth mover's distance as a metric for image retrieval. *IJCV: International Journal of Computer Vision*, 40, 2000.



a)

	0	1	2	3	4	5	6	7	8	9
0	1.000	0.821	0.921	0.823	0.768	0.908	0.896	0.747	0.833	0.879
1	0.821	1.000	0.864	0.622	0.860	0.816	0.720	0.831	0.664	0.720
2	0.921	0.864	1.000	0.731	0.797	0.884	0.833	0.713	0.758	0.810
3	0.823	0.622	0.731	1.000	0.669	0.812	0.945	0.522	0.955	0.948
4	0.768	0.860	0.797	0.669	1.000	0.758	0.756	0.736	0.698	0.744
5	0.908	0.816	0.884	0.812	0.758	1.000	0.874	0.603	0.868	0.865
6	0.896	0.720	0.833	0.945	0.756	0.874	1.000	0.604	0.952	0.987
7	0.747	0.831	0.713	0.522	0.736	0.603	0.604	1.000	0.526	0.585
8	0.833	0.664	0.758	0.955	0.698	0.868	0.952	0.526	1.000	0.957
9	0.879	0.720	0.810	0.948	0.744	0.865	0.987	0.585	0.957	1.000

b)

Figure 3: a) Spatial angles histogram of digits 0 to 9 using $\Psi = \{90, 180\}$. b) Correlation values

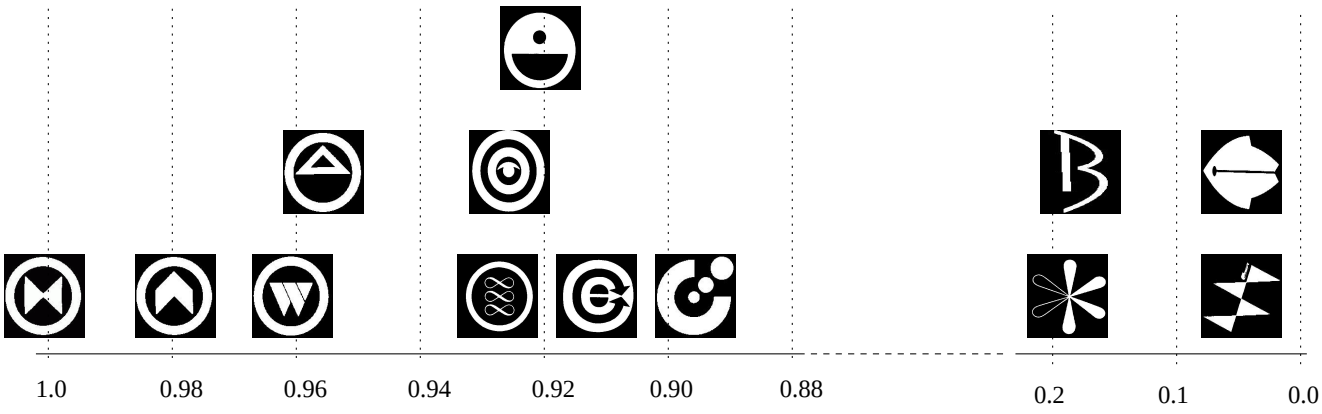


Figure 4: Trademarks example

Affine Transformation Computation of Planar Objects*

Julián Ramos Cózar, Nicolás Guil and Emilio L. Zapata
Dpt. of Computer Architecture
University of Málaga
Málaga, Spain, E-29081

Abstract

In this work a new method to detect the affine transformation that relates two views of a planar object is presented. It is a generalization of a previous work that uses invariant tables generated from contour information to detect the similarity transformations that a planar shape has suffered with respect to a template. Several improvements for invariant table generation and comparison have been carried out in order to reduce the computation complexity of the process. Different search strategies have been implemented and compared for the non-invariant parameters estimation. Experiments performed with images of real objects confirm the applicability of the method.

1 Introduction

Weak perspective is probably the more common approximation to model camera image generation in computer vision. The affine transformation is a more generic 2D→2D projection model and it is a good approximation to the perspective projection under certain conditions [1]. Additionally, there exist many applications where object deformation can be modelled by this kind of transformation. For this reason, it is interesting to establish methods to find the affine matrix that relates two images.

There are several methods to find the transformations between a planar template and an image where this template is included and transformed with arbitrary parameters [5]. The selection of the most suitable method will be based on considerations such as the kind of shapes we are dealing with and the ability to cope with practical problems such as occlusion, noise, etc. A simple approach to object detection is to find, for every possible parameter value, the template transformation that produces a better matching with the image shape. However, the search space can become overwhelmingly large. The search space can be reduced if invariant features are used. Several methods have been proposed to detect the transformation of a planar object based on the Hough Transform (hashing methods). We can take advantage of the Hough Transform's useful properties, such as its relative insensitivity to noise and robustness to occlusions. However, it demands high computational and storage requirements.

In this paper the method presented in [3] for the detection of the similarity transformations will be generalized to cope with the affine transformations. It is based on evidence gathering methods that use invariant contour information in order to reduce the computational complexity of the computation. In previous works [2] this method was successfully used to detect the perspective and orthoperspective projection parameters.

The rest of the paper is organized as follows. In next section, the method presented in [3] to detect the similarity transformations relating two planar shapes will be reviewed. Section 3 introduces the expressions that describe the modifications that points and tangent angles undergo when an affine transformation is applied to a contour. A new method for planar shape detection under affine transformations and the optimizations that allow us to speed-up the process will be shown in Sect. 4. Finally, in Section 5, several implementations of the original algorithm will be tested.

2 Contour Invariants to Similarity Transformations

The contour points of a shape are characterized by the parameters $p = \langle x, y, \theta \rangle$, where x and y are the coordinates of the points in a two-dimensional space and θ is the angle of the gradient vector in the image at this edge point. Two more angles will be used for the contour invariant information generation: the *pairing angle*, ξ, defined as the positive difference that must exists between the angles of two edge points gradient vectors to be paired (i.e., two points, p_i and p_j, will be paired if $\theta_j - \theta_i = \xi$); and the *difference angle*, α_{ij}, defined as the positive angle ($\measuredangle$) formed by the line that joins two paired points, p_i and p_j, and the gradient vector angle of the first point, p_i. All these values are shown in figure 1.

*This work was supported in part by the Ministry of Education and Science (CICYT) of Spain under contract TIC2003-06623.

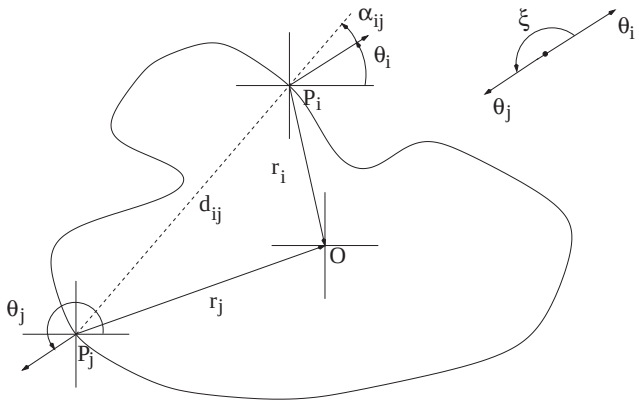


Figure 1: Graphical representation of the angles used in the invariant information generation

From this information, we can derive three contour transformations, based on paired points in the shape edge, that generates new information invariant for some similarity transformations [3]: the $\mathcal{T}$ transformation, $\mathcal{T}(p_i, p_j) = (\theta_i, \alpha_{ij})$; the $\mathcal{S}$ transformation, that includes the distance d_{ij} between the paired points, $\mathcal{S}(p_i, p_j) = (\theta_i, \alpha_{ij}, d_{ij})$; and the $\mathcal{D}$ transformation, $\mathcal{D}(p_i, p_j) = (\theta_i, \alpha_{ij}, \vec{r_i}, \vec{r_j})$, where an arbitrary reference point, $O = (o_x, o_y)$, has to be defined and two vectors are generated from it, $\vec{r_k} = O - p_k, \;\; k = i, j$.

The information generated by the application of these transformations is stored in different tables in order to improve the detection process speed. Thus, the contents of these tables are calculated as follows:

Orientation table ($\mathcal{OT}$). When a pairing is processed by the $\mathcal{T}$ transformation, $\mathcal{OT}[\alpha_{ij}][\theta_i]$ is incremented, indicating how many of the pairings have these (α_{ij}, θ_i) values. The information stored in this table is invariant to scale and displacement. Additionally, the rotation of the shape in a plane causes a circular shift of the corresponding columns of its $\mathcal{OT}$.

Distance table ($\mathcal{ST}$). This is a bidimensional table that uses the information generated by the $\mathcal{S}$ transformation to generate linked lists, in the positions indicated by α_{ij} and θ_i, with the d_{ij} values. Note that this information is invariant to the bidimensional displacement of the shape.

Displacement table ($\mathcal{DT}$). The $\mathcal{D}$ transformation is applied in order to build this table. Linked lists are created for each α_{ij} and θ_i and the data stored at each list position are the $\vec{r_i}$ and $\vec{r_j}$ reference vectors for the points p_i and p_j respectively.

The whole detection process can be segmented in three stages, each one using the information generated in the previous step, as it is shown in Fig. 2:

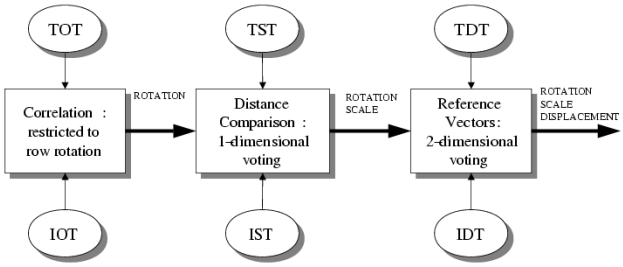


Figure 2: Similarity parameters detection segmentation

- The orientation is computed comparing similar columns in both $\mathcal{TOT}$ (template $\mathcal{OT}$) and $\mathcal{IOT}$ (image $\mathcal{OT}$) tables for different circular shifts. A matching process between both tables is carried out for each shift, generating a voting. The highest value indicates the correct orientation.

- The scale calculation is carried out using the information generated by the $\mathcal{TST}$ and the $\mathcal{IST}$ after eliminating the effect of the previous transformation (a rotation) in the contour of the template. If a different-from-zero position of the $\mathcal{TST}$ coincides with the equivalent one in the $\mathcal{IST}$, distances stored in the linked list associated with this position in the $\mathcal{IST}$ are used to vote in an accumulator. In this operation, each distance value from the $\mathcal{IST}$ is divided by all the distance values in the $\mathcal{TST}$ for this specific (α_{ij}, θ_i) value. The calculated scale is used to vote in a one-dimensional Hough space. The highest peak value will indicate the scale of the image with respect to the template.

- In order to calculate the displacement, the value of the reference vector for each entry of the $\mathcal{TDT}$ – multiplied by the scaling – is added to the coordinates of the paired points stored in the linked list associated with each entry of the $\mathcal{IDT}$. The positions pointed out by these vectors are used to increment a bidimensional space. The maximum position in this accumulator will give us the situation, in the image, of the equivalent reference point defined in the template shape.

3 Contour Modification under Affine Transformations

Given an image $F(x, y)$, we can express the resulting image $G(u, v)$, after applying a generic affine transformation to F, as:

$$u_i = a_{11}\cdot x_i + a_{12}\cdot y_i + b_x, \quad v_i = a_{21}\cdot x_i + a_{22}\cdot y_i + b_y, \quad (1)$$

where the transformation matrix **A** given by this expression must be invertible.

The algorithms presented here transform the gradient angle information in order to reduce the computational complexity. Thus, it is interesting to obtain the relationship between the gradient angle of an image shape, $F(x, y)$, and an affine transformation of this image, the image $G(u, v) = F(u(x, y), v(x, y))$. Taking into account that the variables u and v depend on x and y, as shown in (1), if the first derivative function is applied we get:

$$\tan\theta_A = \frac{a_{21} + a_{22}\cdot\tan\theta}{a_{11} + a_{12}\cdot\tan\theta}, \quad (2)$$

where θ_A and θ are the values of the tangents in the affine transformed image and in the original image, respectively.

An important consequence of expression (2) is the following lemma:

> ***Lemma**: Let $\mathcal{C}$ be a contour whose points are paired for the invariant tables generation using the* pairing angle $\xi = 180^{\circ}$. *If an affine transformation is applied to the points in this contour, the original pairings are kept with the same* pairing angle, $\xi_T = 180^{\circ}$.

4 Detection of the Affine Transformation Parameters

The method for the detection of the similarity transformations will be generalized to the affine transformations group in this section. First, we need to establish a fixed sequence of basic transformations, equivalent to any affine transformation. Then, the method described in Sect. 2 will be generalized for this chain of operations. Finally, several optimizations in table generation and non-invariant parameters search will be presented.

4.1 Affine Transformation Decomposition

An affine transformation can be defined by means of a matrix **A**. Using the classical algebra results, we can express a real matrix **A** as the following product:

$$\mathbf{A} = \mathbf{T}(\vec{D}) \cdot \mathbf{E_u}(s) \cdot \mathbf{R}(\beta) \cdot \mathbf{E_{nu}}(\vec{S}) \cdot \mathbf{Sh_x}(k_x), \quad (3)$$

where $\mathbf{T}$ represents a translation $\vec{D}$; $\mathbf{E_u}$ an uniform scaling s; $\mathbf{R}$ a rotation of β degrees; $\mathbf{E_{nu}}$ an y-axis scaling s_{yx}; and $\mathbf{Sh_x}$ an x-axis shear k_x. Note that only the first two transformations –shear and y-axis scaling– introduce a distortion in the original shape. The rest are similarity transformations, changing either the position, the orientation or the size of the distorted shape. Thus, if the effect of the first two transformations could be eliminated, the method based on invariant tables presented in Sect. 2 for similarity transformations computation could be applied.

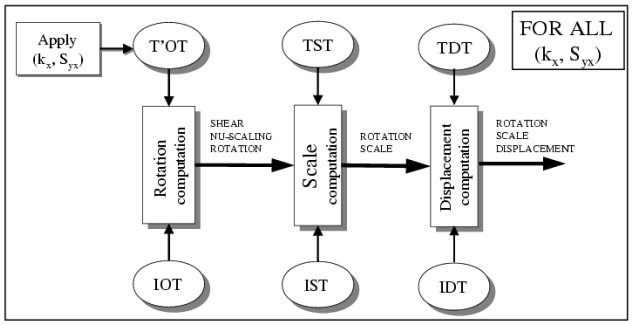


Figure 3: Block diagram of the detection method

Normally, the number of edge points in the template is lower than in the image, where several different shapes can appear. Thus, we will consider that these transformations are applied to a template (or reference) image in order to obtain its instance in an test image.

4.2 Detection Process

As it is shown in expression (3), the effects of the shear k_x and non-uniform scaling s_{yx} must be applied to the template contour in order to use the method to detect similarity transformations. All possible (k_x, s_{yx}) pairs must be tested, as they are unknown. The template orientation tables will be built from the deformed contour with these parameters and the associated rotation will be estimated by the correlation with the image orientation table. After eliminating the effects of these parameters, the scale and the displacement can be computed using the rest of invariant tables. The set of parameters that maximices the similarity between the transformed template contour, using the detected parameters, and the image contour will be considered the solution. This process is summarized in Fig. 3. The similarity σ will be measured using the expression:

$$\sigma(\mathcal{T}, \mathcal{I}) = \frac{1}{n_T}\sum \exp\left(-0.1\sqrt{\Delta_x^2 + \Delta_y^2}\right)\cdot\left|\cos\left(\varepsilon(x, y)\right)\right|, \quad (4)$$

where the summation is over all pixels on the template contour, n_T is the number of pixels on the template contour, (Δ_x, Δ_y) is the displacement –in pixels– from the edge point (x, y) in the template image to the nearest edge point in the image, and $\varepsilon(x, y)$ is the difference angle between the

gradient direction of the template at (x, x) and the gradient direction at the nearest image edge. Note that σ will always be in the range from 0 (very different images) to 1 (very similar images).

4.3 Invariant Tables Modification

In a general situation, where the k_x and s_{yx} values are unknown, invariant tables must be generated and compared from the transformed contour for each (k_x, s_{yx}) possible combination using the expressions (1) and (2). This process require a high computational complexity. However, several improvements have been carried out in table generation to obtain a good performance.

Template $\mathcal{OT}$ building is based on both the calculation of the template gradient angle, θ, and the difference angle, α, of contour paired points. In order to reduce its computation time, we have studied the modifications in an angle value that arise under an affine transformation. This allows us to generate an $\mathcal{OT}$ for a specific (k_x, s_{yx}) deformation by directly transforming the original template $\mathcal{OT}$, when it was created using a paring angle of $\xi = 180^\circ$. Particularizing expression (2) for the affine transformation with a x-axis shear, k_x, and a non-uniform y-axis scaling, s_{yx}, we get:

$$\theta_T = \arctan\left(\frac{s_{yx} \cdot \tan\theta}{1 + k_x \cdot \tan\theta}\right). \qquad (5)$$

Similarly, an expression for the transformation of the difference angle when a shear and a non-uniform scaling are applied to a shape can be obtained:

$$\alpha_T = \arctan\left(\frac{s_{yx} \cdot \tan(\theta + \alpha)}{1 + k_x \cdot \tan(\theta + \alpha)}\right) \measuredangle\ \theta_T. \qquad (6)$$

Thus, the generation of a transformed template $\mathcal{OT}$ for any (k_x, s_{yx}) applied to it from its original $\mathcal{OT}$ can be performed by gathering the votes in $\mathcal{OT}[\alpha][\theta]$ to the new location $\mathcal{OT}_T[\alpha_T][\theta_T]$. This process is very efficient since this table is very sparse and can be accelerated by using precomputed transformation tables.

4.4 Search Strategies

The search for the unknown parameters k_x and s_{yx} can be accomplished in different ways. The more precise solution will be found if we implement an exhaustive search. However, we can perform this task in a more efficient manner using a multipass approach and non-deterministic optimization algorithms.

In the *multipass* approach, program execution can be accelerated by reducing the search space size. Starting with a coarse estimation of the parameters, successive steps focus on a narrower interval around the solution estimated in the previous step. A good initial coarse estimation is computed using a resolution of 0.2. Subsequent steps use a resolution of 0.1 and 0.05.

A *stochastic* search, UEGO (*Universal Evolutionary Global Optimizer*) [4], based on evolutive algorithms has also been implemented. More precisely, UEGO searchs for groups of solutions and these solutions are optimized using a Single Agent Stochastic Search method (SASS).

Another important reduction in the computational complexity can be achieved if the maximization criterion is the invariant template and image orientation tables matching, measured as its maximum normalized correlation. An immediate consequence of using this criterion is the lost of accuracy in the detection process. In this situation, all possible (k_x, s_{yx}) pairs should be tested in order to find the applied rotation. The scale and displacement –which are the most computationally expensive tasks– will be computed only once at the end of the process, after eliminating the effect of the other parameters in the template contour.

5 Experimental Results

We have used different images in order to test the behavior for different implementations of the whole detection process. Tables 1 and 2 summarizes the most important accuracy (σ, in %) and computational time (T, in seconds) results for the deterministic and stochastic search strategies respectively. The accuracy of the detection process is checked by using the function σ given by expression (4) and by superimposing the transformed template contour onto the image (second column of Tbl. 1). The number of experiments whose σ values are greater than $0.9{\cdot}\sigma_{\mathrm{Max}}$ is indicated (in %) by σ_{90}. Times are given for executions in a SGI Workstation with an R–10K@225 MHz processor. The light gray color indicates the non-precise detections.

The similarity between contours (Contour) and the matching between invariant tables (InvTables) was used as optimization criterion in both parameter search. The version of the algorithm which directly transform the Template Orientation Table was named -OT. For the deterministic search, the Sequential and Multipass approach were implemented for comparison.

6 Conclusions

A new method to detect the affine transformation that relates two views of a planar object has been presented. Several optimizations improve the computational complexity of the process. Experiments performed with images of real objects have shown the performance of several implementations of

the algorithm. Highest computational times are required for the more precise implementations.

References

[1] J.Y. Aloimonos. Perspective approximations. *Image and Vision Computing*, 18(3):179–192, August 1990.

[2] J.R. Cózar, N. Guil, and E.L. Zapata. Detection of arbitrary planar shapes with 3D pose. *Image and Vision Computing*, 19(14):1057–1070, December 2001.

[3] N. Guil, J.M. González-Linares, and E.L. Zapata. Bidimensional shape detection using an invariant approach. *Pattern Recognition*, 32(6):1025–1038, June 1999.

[4] P.M. Ortigosa. *Métodos estocásticos de optimización global. Procesamiento paralelo*. PhD thesis, Universidad de Málaga, 1999.

[5] Isaac Weiss. Review. Geometric invariants and object recognition. *International Journal of Computer Vision*, 10(3):207–231, June 1993.

Table 1: Results obtained with a deterministic search of the solution

			Contour		InvTables	
Template	Visual	Algorithm	σ	T_{Tot}	σ	T_{Tot}
		Sequential	95.65	11.15	95.65	2.55
		Multipass	92.45	1.21	92.72	0.27
		Sequential-OT	23.97	9.18	24.38	0.70
		Multipass-OT	58.54	0.96	25.05	0.07
		Sequential	84.54	9.01	85.36	1.98
		Multipass	84.54	0.93	68.95	0.20
		Sequential-OT	22.76	7.70	85.36	0.60
		Multipass-OT	85.36	0.80	68.95	0.07
		Sequential	87.91	29.41	7.61	7.59
		Multipass	87.91	3.08	24.56	0.79
		Sequential-OT	87.91	24.13	29.64	2.14
		Multipass-OT	87.91	2.54	33.10	0.24
		Sequential	75.98	314.00	55.96	94.68
		Multipass	75.98	32.76	55.96	9.65
		Sequential-OT	73.62	235.27	53.89	23.28
		Multipass-OT	73.62	24.58	53.89	2.63

Table 2: Results obtained with a stochastic search of the solution

Template	Algorithm	σ_{Max}	σ_{Min}	σ_{90}	T_{Mean}	T_{Min}	T_{Max}
	Contour	96.10	65.08	60	3.77	3.34	4.23
	Contour-OT	88.32	17.47	5	3.12	2.76	3.48
	InvTables	96.41	19.15	75	1.78	1.64	2.06
	InvTables-OT	35.97	9.45	15	0.49	0.44	0.53
	Contour	86.85	73.56	75	5.38	4.39	6.99
	Contour-OT	86.65	72.77	75	2.82	2.39	3.11
	InvTables	83.78	72.54	85	1.44	1.29	1.56
	InvTables-OT	86.10	76.36	95	0.48	0.39	0.51
	Contour	89.70	83.64	100	10.15	8.27	12.33
	Contour-OT	89.65	84.10	100	8.56	7.37	9.78
	InvTables	57.39	15.63	5	5.29	4.65	5.67
	InvTables-OT	81.91	16.52	35	1.54	1.34	1.74
	Contour	74.67	69.10	100	110.74	98.85	121.06
	Contour-OT	73.63	68.16	100	85.74	72.91	98.38
	InvTables	73.02	34.15	20	67.26	59.30	74.21
	InvTables-OT	63.58	52.56	35	20.10	18.71	21.91

Multiscale analysis and shape representation

Luis Alvarez, Luis Mazorra and Francisco Santana

Departamento de Informática y Sistemas

Universidad de Las Palmas de Gran Canaria

Campus de Tafira, 35017, Las Palmas

{lalvarez,lmazorra,fsantana}@dis.ulpgc.es

Abstract

In this paper, we present a geometric invariant shape representation using morphological multiscale analyses. The geometric invariant is based on the area and perimeter evolution of the shape under the action of a morphological multiscale analysis. In the case of similarity transformations, the proposed geometric invariant is based on a scale-normalized evolution of the isoperimetric ratio of the shape. In the case of general affine geometric transformations the proposed geometric invariant is based on a scale-normalized evolution of the area. We present some numerical experiments to evaluate the performance of the proposed models.

1 Introduction.

Shape representation methods play an important role in systems for object recognition and analysis. Shape description refers to the methods that result in a numeric descriptor of the shape and could be a step subsequent to shape representation. Another classification of shape analysis methods is based on the use of shape boundary points as opposed to the interior of the shape. The two resulting classes of methods are known as boundary (also called external) and global (also called internal), respectively.

In the last years, multiscale analyses have became a common tool for many tasks in computer vision. A multiscale analysis can be defined as an operator $T_t(f)$ which provides for an original image f a sequence of images $T_t(f)$ which represent the image at a coarse scale t.

In this paper we deal with morphological multiscale analyses, which satisfy the morphological invariance, that is, the multiscale analysis $T_t(f)$ commutes with any increasing histogram modification of the image. It means that for any increasing function $g(.)$

$$T_t(f) \circ g = T_t(f \circ g).$$

The underlying hypothesis associated to this morphological invariance is that the contrast between the different objects present in the image is not important at all, and that all the information present in the image is described by the geometry of the level sets of the image. In particular, the way a shape changes under the action of a morphological multiscale analysis depends only on the geometry of its boundary.

The main underlying idea we propose in this paper is that if we take any global invariant of a shape and we follow the evolution of such invariant under the action of a morphological multiscale analysis then, this evolution is also un invariant of the shape, but it contains much more robust and discriminant information of the geometry of the shape that just the invariant for the initial shape. In particular, we propose in this paper to use the evolution of the area and/or perimeter of the shape across the scales under the action of different morphological multiscale analyses as basic tools to find out scale-space global shape representation. The main advantage of the morphological multiscale analyses with respect to the classical linear-scale space is that the evolution of the shape depends just on the geometry of the shape and it is not depends at all on the contrast of the shape with respect to the background or the relative location of other shapes presented in the image which is not the case in the linear scale-space where the way a shape evolves depends on the contrast and location with respect to other shapes presented in the image.

As it was proved in [2],[1] under some minimal architectural assumptions, all the morphological and similarity invariant multiscale analyses are generated by the partial differential equation:

$$\frac{\partial u}{\partial t} = \beta(\ curv(u))\ \|\nabla u\|\,, \tag{1}$$

where $curv(u)(x,y)$ is the curvature of the level line

passing by the point (x, y), that is:

$$curv(u) = div\left(\frac{\nabla u}{\|\nabla u\|}\right). \tag{2}$$

and $\beta(.)$ is given by

$$\beta(s) = \begin{cases} \beta(1)s^p & if \quad s \geq 0 \\ \beta(-1)\,(-s)^p & if \quad s < 0. \end{cases} \tag{3}$$

where $p > 0$. Therefore the model depends on 3 parameters, p, β_{-1}, β_1. If $u(t, x, y)$ is the solution of equation (1), for the initial datum f, then

$$u(t, x, y) = T_t(f)(x, y).$$

Following the morphological principle, we will consider that a shape S_0 is given by a level set of the image f, that is:

$$S_0 = \overline{\{(x, y) :\ f(x, y) < \lambda\}},$$

for some λ, where for a set A, we denote by $\overline{A}$ the closure of A, that is, the minimum closed set including A. We will denote by $S(t)$ the evolution across the scales of S_0, that is:

$$S_{(p,\beta_{-1},\beta_1)}(t) = \overline{\{(x, y) :\ T_t(f)(x, y) < \lambda\}}.$$

We will also denote by $C_{(p,\beta_{-1},\beta_1)}(t)$ the boundary of $S_{(p,\beta_{-1},\beta_1)}(t)$. For the case $C_{(p,\beta_{-1},\beta_1)}(t)$ is a family of single Jordan curves, we can interpret the evolution of $C_{(p,\beta_{-1},\beta_1)}(t)$ in terms of curve evolution. In fact, $C_{(p,\beta_{-1},\beta_1)}(t)$ is a solution of the curve evolution equation

$$\frac{\partial C_{(p,\beta_{-1},\beta_1)}}{\partial t} = \beta(k)\vec{N}, \tag{4}$$

where $\vec{N}$ represents the unit inward normal direction to the curve $C_{(p,\beta_{-1},\beta_1)}(t)$ and k is the curvature. In the last years, a lot research have been devoted to this curve evolution equation see, for instance, [3], [4], [6], [7], [8], [2], [5].

The organization of the paper is as follows: In section 2, we analyze the similarity invariant shape representation, and we propose as geometric invariant a scale-normalized isoperimetric ratio evolution. In section 3, we study the affine invariant shape representation, and we propose as geometric invariant a scale-normalized area ratio evolution. In section 4, we present some numerical experiments.

2 The scale-normalized isoperimetric ratio evolution.

We will use as similarity invariant of a bounded shape S_0 the scale-normalized isoperimetric ratio evolution $I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t})$ given by the following definition

Definition 1 *Let S_0 be a bounded shape. We define the scale-normalized isoperimetric ratio evolution $I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t})$ as the function*

$$I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t}) = 4\pi \frac{\left|S_{(p,\beta_{-1},\beta_1)}(\tilde{t}\sqrt{|S_0|})\right|}{\left|C_{(p,\beta_{-1},\beta_1)}(\tilde{t}\sqrt{|S_0|})\right|^2}.$$

We notice that $I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t}) \leq 1$, and $I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t}) = 1$ only for the case when $S_{(p,\beta_{-1},\beta_1)}(\tilde{t}\sqrt{|S_0|})$ is a circle. Next, we will show that $I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t})$ is a similarity invariant of the shape S_0.

Theorem 1 *Let $T_t^{(p,\beta_{-1},\beta_1)}(f)$ be a morphological multiscale analysis invariant under similarity transformations, S_0, S_0' be two bounded shapes such that there exists a similarity transformation H with $H(S_0') = S_0$, Then:*

$$I^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t}) = I^{S_0'}_{(p,\beta_{-1},\beta_1)}(\tilde{t}) \ \ for\ \tilde{t} \geq 0.$$

Proof: See [1]

3 Morphological Affine Invariant Representation of a Shape.

We consider a general affine transformation given by

$$H(x, y) = A\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix}$$

where A is a 2×2 matrix with $|A| \neq 0$.

[2] show that the only affine invariant morphological multiscale analysis is given by

$$\beta(s) = \begin{cases} \beta_1 s^{\frac{1}{3}} & if \quad s \geq 0 \\ \beta_{-1}(-s)^{\frac{1}{3}} & if \quad s < 0, \end{cases}$$

where $\beta_1 \geq 0$ and $\beta_{-1} \leq 0$. In this case we have that

$$H\left(T_{t'(H,t)}(f)\right) = T_t(H(f)),$$

where

$$t'(H, t) = |A|^{\frac{4}{3}}\, t.$$

On the other hand, given two bounded shapes S_0, S_0', such that there exists an affine transformation H with $H(S_0') = S_0$, we have that:

$$\left|S_{(p,\beta_{-1},\beta_1)}(\tilde{t})\right| = \frac{\left|S'_{(p,\beta_{-1},\beta_1)}(\sqrt{|A|}\tilde{t})\right|}{|A|} \quad for\ any\ t \geq 0. \tag{5}$$

One of the main advantages of our approach is that we use a two parameters family of affine invariant scale spaces. In our knowledge we are the first to enjoy of such possibility, usually people take directly the classical case which correspond to $\beta_1 = -\beta_{-1}$. With this two parameters family, we can get much more information about the shape geometry that just using the classical affine invariant multiscale analysis.

In the case of the affine invariant representation, we can not use the scale-normalized isoperimetric ratio because the perimeter is not invariant under affine transformations. We propose a geometric invariant based just on the area evolution. However we note that we could use any other global affine invariant of the shape.

Next, We will introduce the scale-normalized area ratio.

Definition 2 *For a bounded shape S_0, we define the scale-normalized area ratio evolution $AR^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t})$ as the function*

$$AR^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t}) = \frac{\left|S_{(p,\beta_{-1},\beta_1)}(\tilde{t}\sqrt{|S_0|})\right|}{|S_0|}.$$

Next, we will show that $AR^{S_0}_{(p,\beta_{-1},\beta_1)}(\tilde{t})$ is an affine invariant of the shape S_0.

Theorem 2 *Let $T_t^{(\frac{1}{3},\beta_{-1},\beta_1)}(f)$ be a morphological multiscale analysis invariant under affine transformations ($p = \frac{1}{3}$), S_0, S_0' be two bounded shapes such that there exists an affine transformation H with $H(S_0') = S_0$, Then:*

$$AR^{S_0}_{(\frac{1}{3},\beta_{-1},\beta_1)}(\tilde{t}) = AR^{S_0'}_{(\frac{1}{3},\beta_{-1},\beta_1)}(\tilde{t}) \;\; for\ \tilde{t} \geq 0.$$

Proof: See [1]

4 Numerical experiments

The numerical algorithms that we use to implement numerically the morphological multiscale analysis are based on the techniques studied in [3]. We use a simple explicit finite difference scheme to discretize equation (**??**). We have focussed our attention on the qualitative behavior of the proposed models and we have not devoted a lot of time to study the efficiency of the numerical algorithms. Of course, we could use more efficient algorithms to estimate the shape evolution like accurate curve evolution algorithms for equation (**??**). However the application of these curve evolution type algorithms could be delicate in some cases due to, on the one hand, the boundary of a shape could be defined by several curves (in the case the shape has holes) and on the other hand, as we are going to see, the evolution of the shape under the action of a morphological multiscale analysis can develop singularities in the shape evolution, that is the shape can be split in several shapes, two boundary curves can touch each other and become a single curve, etc...

Next, we will present some experiments using the scale normalized isoperimetric ratio evolution $I^{S_0}_{(1,\beta_{-1},\beta_1)}(\tilde{t})$. We will use some synthetic shapes given in figure 1. All the shapes (except the circle) have similar initial isoperimetric ratio (in fact theoretically the isoperimetric ratio is exactly the same for all shapes. However, in practice, because of pixel noise and numerical errors, the computed isoperimetric ratio is not the same), therefore the isoperimetric ratio for the initial shapes, is not useful at all to classify this synthetic shape database. However, as we are going to see, when we follow the evolution of the isoperimetric ratio under the action of a morphological multiscale analysis we can discriminate easily between the different shapes. The shapes are organized as follow: For each shape we have evaluated a similarity transformation where we have rotated and changed the size of the original shape. So shapes $1-2$, $3-4$, $4-5$, $5-6$ and $7-8$ are equivalent modulus a similarity transformation. Shape 9 is similar to shape 7 but in shape 9, we have changed the location of the inside square. We will compare $I^{S_0}_{(1,\beta_{-1},\beta_1)}(\tilde{t})$ for the different shapes for $\tilde{t} \in [0, 0.3]$. We recall that

$$\lim_{\tilde{t}\to\left(\sqrt{\frac{1}{\pi}}\right)^-} I^{S_0}_{(1,-1,1)}(\tilde{t}) = 1.$$

Therefore $\tilde{t} = \sqrt{\frac{1}{\pi}} \simeq 0.56$ is the upper bound for the scale comparison. In practice, we are not interested in taking this upper bound as final scale because the isoperimetric ratio is going to be close to 1 for any shape when we approach the upper bound scale and it is not discriminant from a geometric point of view. In the experiment we present we have taken the final scale equal to $0,3$ which seems a reasonable choice, however we have not studied how to optimize the choice of the final scale and we have not tested different final scales.

In figure 2, we present the evolution of $I^{S_0}_{(1,-1,1)}(\tilde{t})$ for the shapes of figure 1. First, we notice that the similarity invariant is very well preserved because the graphs of the similarity equivalent shapes evolves very close each other. We observe that at the initial scales the pixel noise introduced in the discrete representation of the synthetic shapes produces some perturbations in the isoperimetric ratio estimation, however, we can realize that when we move across the scales these initial perturbations disappear which is a very good behavior. We can also realize that we can discriminate very well between the different shapes following the isoperimetric ratio evolution. We can observe that the isoperimetric ratio of the shapes converges to 1 as it was shown in [6] We notice that shapes $7-8-9$ have similar evolutions because the evolution of $I^{S_0}_{(1,-1,1)}(\tilde{t})$ is not altered

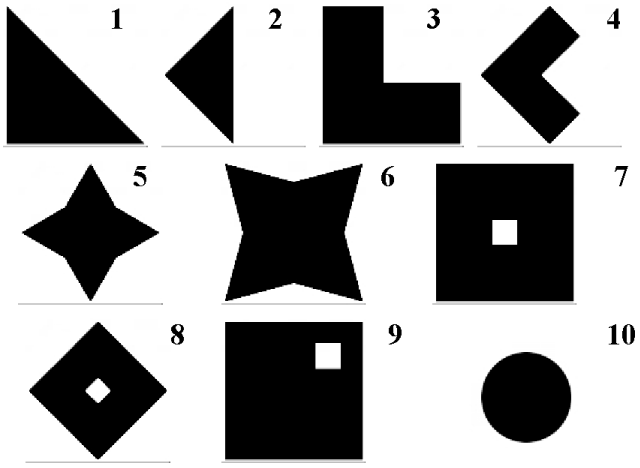


Figure 1: Test shapes used to evaluate the scale-normalized isoperimetric ratio.

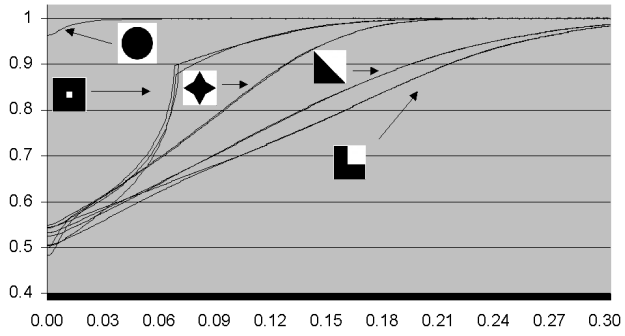


Figure 2: Evolution of $I^{S_0}_{(1,-1,1)}(\tilde{t})$ for the shapes of figure 2.

by the location of the inside square. We can observe that, since the curvature of the contour of the hole goes to infinity at the scale where the hole vanishes, a singularity (a point where the evolution is not smooth) appears at such scale. In fact, in some way, we could "characterize" the holes of the shapes following the singularities of the isoperimetric ratio evolution, but the studying of such behavior is beyond the scope of this paper.

In figure 3, we present the evolution of $I^{S_0}_{(1,0,1)}(\tilde{t})$ for the shapes of figure 1. We notice that in this case, shapes 7 and 9 have different evolution following the location of the inside square. This behavior is illustrated in figure 4 where we show some steps of the evolution of $S_{(1,0,1)}(t)$ across the scales for shape 7 and 9. We observe that shape 7 splits in 4 different shapes when the exterior contour touches the inside square. For shape 9 we observe that a singularity in the isoperimetric ratio evolution appears at the scale where the exterior contour touches the inside square and the hole

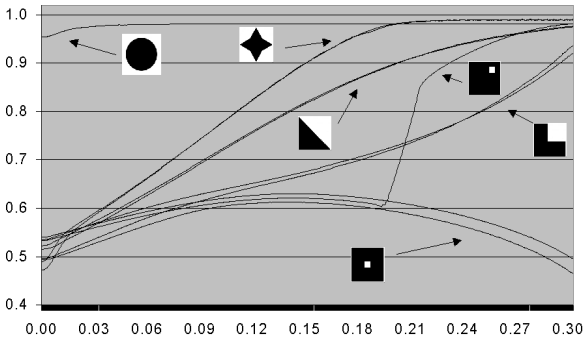


Figure 3: Evolution of $I^{S_0}_{(1,0,1)}(\tilde{t})$ for the shapes of figure 2.

disappears (two boundary curves become a single one). On the other hand, looking at the evolution of shapes $7, 8$ we can observe that in this case the isoperimetric ratio does not converge to 1 as in the case of the mean curvature motion evolution.

In figure 5, we present the evolution of $I^{S_0}_{(1,-1,0)}(\tilde{t})$ for the shapes of figure 1. In the appendix we will show that the asymptotic state of the shape for this multiscale analysis is the convex-hull of the initial shape, so, in particular, the isoperimetric ratio converges towards the isoperimetric ratio of the convex-hull of the shape.

Next, we will present some experiments for the scale normalized area ratio evolution $AR^{S_0}_{(\frac{1}{3},\beta_{-1},\beta_1)}(\tilde{t})$. In this case, we want to discriminate shapes following general affine transformations, so we will use a synthetic shape database composed by affine equivalent shapes. This collection of synthetic shapes is presented in figure 6. For each shape we have evaluated an affine transformation where we have changed the horizontal and vertical sizes in a different way. So shapes $1-2$, $3-4$, $4-5$, $5-6$, $7-8$ and $9-10$ are equivalent modulus an affine transformation. We will compare $AR^{S_0}_{(\frac{1}{3},\beta_{-1},\beta_1)}(\tilde{t})$ for the different shape for $\tilde{t} \in [0, 0.3]$.

In figure 7, we present the evolution of $AR^{S_0}_{(\frac{1}{3},-1,1)}(\tilde{t})$ for the shapes of figure 6. Each shape has associated two graphs which correspond to the evolution of $AR^{S_0}_{(\frac{1}{3},-1,1)}(\tilde{t})$ for the different transformations of the shape. We can observe that initially the area ratio is always equal to 1 and it decreases across the scales. We note that the affine invariance of the multiscale analysis is very well preserved, it means that the evolution of two affine equivalent shapes go so close that most of the time seems to be a single graph in figure 7.

In figure 8, we present the evolution of $AR^{S_0}_{(\frac{1}{3},0,1)}(\tilde{t})$ for the shapes of figure 6. We notice that in this case, only the convex region of the shape evolves, so this behavior

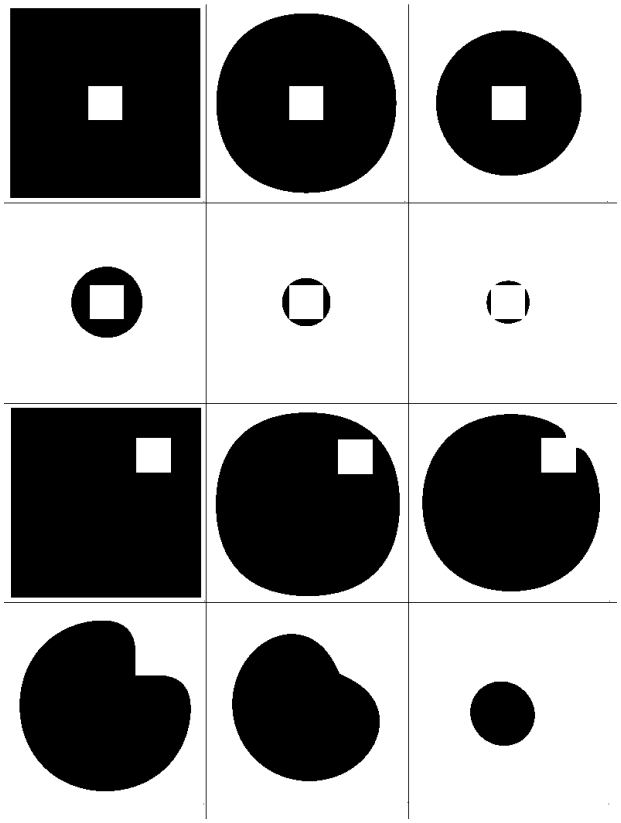


Figure 4: From left to right and from top to down: Evolution of $S_{(1,0,1)}(\tilde{t})$ for shapes 7 and 9 of figure 2.

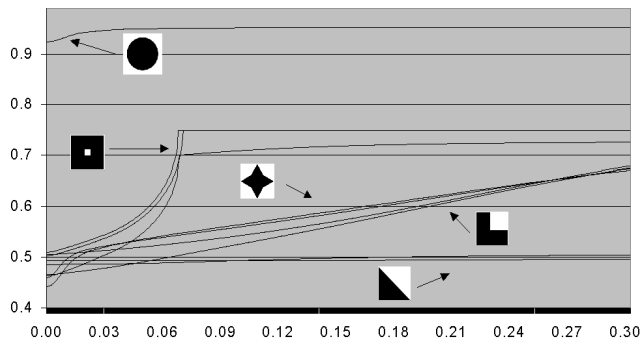


Figure 5: Evolution of $I^{S_0}_{(1,-1,0)}(\tilde{t})$ for the shapes of figure 2.

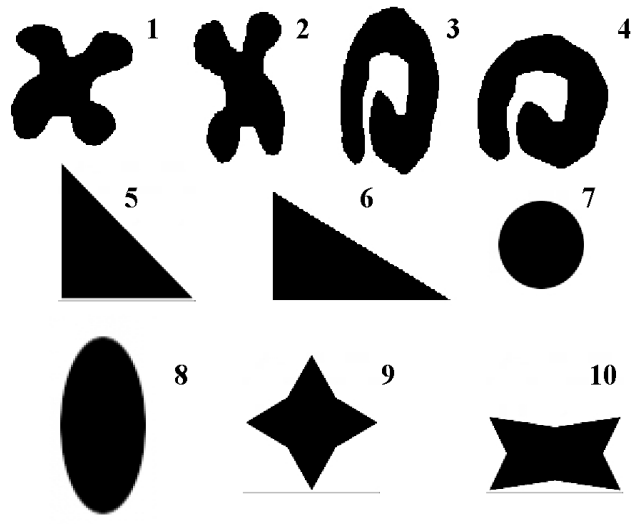


Figure 6: Test shapes used to evaluate the affine invariant scale-normalized area ratio evolution

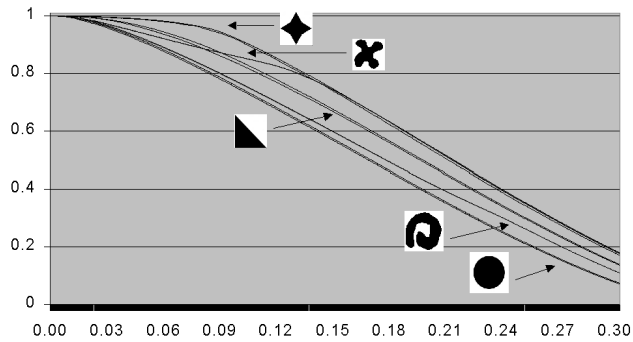


Figure 7: Evolution of $AR^{S_0}_{(\frac{1}{3},-1,1)}(\tilde{t})$ for the shapes of figure 7.

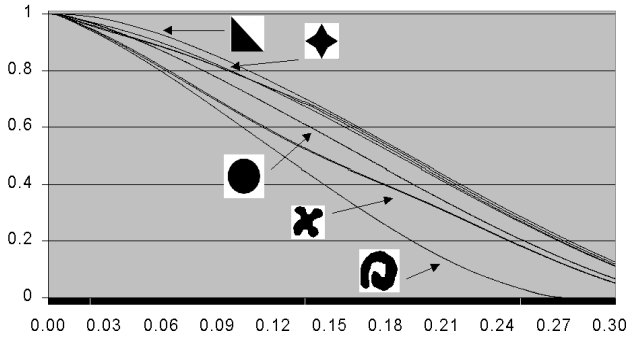


Figure 8: Evolution of $AR^{S_0}_{(\frac{1}{3},0,1)}(\tilde{t})$ for the shapes of figure 7.

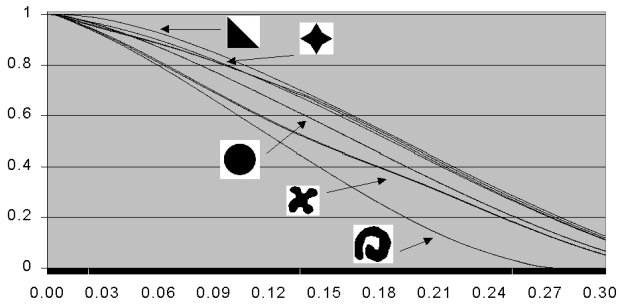


Figure 9: Evolution of $AR^{S_0}_{(\frac{1}{3},-1,0)}(\tilde{t})$ for the shapes of figure 7.

produces a strong discrimination between the evolution of shapes following the geometry of their convex and concave regions. This effect can be observed if we compare the evolution of shapes 3 and 7. The evolution of $AR^{S_0}_{(\frac{1}{3},0,1)}(\tilde{t})$ for these two shapes is very different, but the evolution of $AR^{S_0}_{(\frac{1}{3},-1,1)}(\tilde{t})$ for the same shapes are much more similar. So in practice, it means that using the information of the area evolution with different values of β_{-1} and β_1 we obtain a better discrimination power between different shapes.

In figure 9, we present the evolution of $AR^{S_0}_{(\frac{1}{3},-1,0)}(\tilde{t})$ for the shapes of figure 6. The evolution with the multiscale analysis $T_t^{(\frac{1}{3},-1,0)}$ is more sensitive to pixel noise than the ones corresponding to $\beta_1 > 0$. The reason is that in this case we do not have a regularization effect on the boundary. For instance the evolution of the triangles given by shapes 5 and 6 are quite different because of some pixel errors introduced by the application of the affine transformation to shape 5. The regularization effect on the boundary is a well-known property of the morphological multiscale analysis, it means that the multiscale analysis smooths the contours, by lowering the curvature value (see for instance [4] for more details).

Acknowledgments. This work has been partially supported by the spanish research project TIC 2000-0585.

References

[1] Alvarez L., Blanc A-P., Mazorra L. and Santana F., *Geometric Invariant Shape Representations Using Morphological Multiscale Analysis*, Journal of Mathematical Imaging and Vision, Vol 18 pp 145-169., 2003

[2] Alvarez L., Guichard F., Lions P.-L. and Morel J-M., *Axioms and fundamental equations in image processing*, Arch. Rational Mech. Anal., Vol. 123, 199–257, 1993.

[3] Alvarez L. and Morel J-M., *Formalization and Computational Aspects of Image Analysis*, Acta Numerica pp. 7-59, 1994.

[4] Angenent S., *Parabolic equations for curves on surfaces, Part I. Curves with p-integrable curvature*, Annals of Mathematics Vol. 132, 451-483, 1990.

[5] Angenent S., Sapiro G. and Tannenbaum A., *On the affine heat equation for nonconvex curves*, Journal of the American Mathematical Society Vol. 11, no. 3, 601–634, 1998.

[6] Gage M., *An isoperimetric inequality with applications to curve shortening*, Duke Mathematical Journal , Vol. 50, No. 4, 1225-1229, 1983.

[7] Gage M., *Curve shortening makes convex curves circular*, Inventiones Mathematicae, Vol. 76, 357-364, 1984.

[8] Gage M. and Hamilton R.S., *The heat equation shrinking convex plane curves*, J. Differential Geometry, Vol. 23, 69–96, 1986.

[9] Grayson M. , *The heat equation shrinks embedded plane curves to round points*, J. Differential Geometry, Vol 26, 285–314, 1987.

A Model Based Edge Location with Subpixel Precision

A. Trujillo, L. Alvarez, J. Esclarin
Dpto. Informatica y Sistemas
Universidad de Las Palmas de Gran Canaria
atrujillo,lalvarez,jesclarin@dis.ulpgc.es

Abstract

Estimation of edge features in an image (sub-pixel position, direction, curvature and change in intensity at both sides of the edge) starting from the calculus of the gradient vector in each pixel is not exact, even in ideal images. In this paper we present a new edge detector based on an edge and adquisition model that does not assume continuity in the image values. Edge features are obtained with high precision using this detector, even in noisy images. When noise is too high, we also propose an iterative method to enhance the image quality, and keep precision in the features obtained. Finally some synthetic and real examples are shown.

1. Introduction

1.1. Previous Work

Edge detection is one of the low level tasks more important in image processing, as a previous stage to estimate other high level information. Most of previous work are focused on a pixel level. However, when higher precision is required, sub-pixel information is needed. Some works in this field has been done by Braggins [1], Naidu [4] and Trujillo [8].

On the other hand, adquisition process adds noise to the image values. Therefore, a smoothing process is needed in order to reduce noise. Work in this field can be classified in methods that assume a known noise model, and those that does not consider any previous model. Important contributions in the first group are the works of Rosenfeld & Kak [6], based on the properties of Fourier transform, and the work of Castleman [2] based on Wiener filters. In the second group are the works of Perona & Malik [5] about anisotropic diffusion, and the work of Lindeberg [3] about multiscale analysis theory.

1.2. Proposed hypothesis

Most of works in literature assume that a digitized image F is the sampled version of a function $f(x, y)$ over a rectangular grid of pixels. In order to apply differential calculus techniques on the image, it is assumed also that f is continuous and differentiable inside its domain. We will use a different hypothesis.

We will assume that an edge is a discontinuity in the intensity values of the function f. Besides, if an edge is crossing over the pixel (i, j), the intensity acquired in that pixel will be:

$$F_{i,j} = \frac{AS_A + BS_B}{h^2}$$

where A and B are the intensities at both sides of the edge, S_A and S_B are the region areas filled by both intensities respectively inside the pixel, and h is the lenght of a pixel side.

We can use a traditional mask to compute partial derivatives, like the ones proposed by Sobel or Prewit [7], in order to look for the maximum values along the gradient direction to detect what pixels belong to an edge. However, with the gradient value obtained is not possible to obtain the exact value of the direction of the edge, nor the change of intensity at both sides. In the next section we propose a new method to estimate these features with higher precision.

2. Edge detection

Let us consider that in the neighborhood of a pixel (i, j) there exists a straight edge with slope between 0 and 1.This edge is represented by the equation $y = a + bx$ that divides the plane in two regions of intensity A and B. Assuming that a, b, A, B are unknown, the value of neigbour pixels are shown in figure 1a, where pixels drawn with light grey color are those that could have an intermediate value between A and B.

Let us consider a $5x3$ window centered on the pixel (i, j). Let us be S_M the sum of the 5 pixel values in the middle column of the window. This value will be:

$$S_M = 5B + \frac{A - B}{h^2} M$$

where M is the area inside the column under the

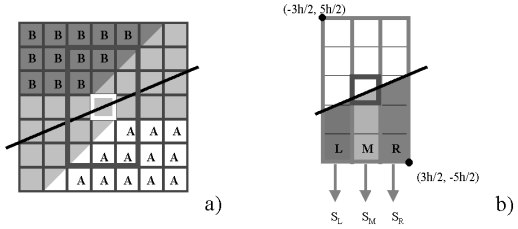


Figure 1:

straight line (see figure 1b). This value is given by:

$$M = \int_{-h/2}^{h/2} (a + bx + 5h/2)\, dx$$

Using the equation of S_M we could obtain the expression of the coefficient a as follows:

$$a = \frac{2S_M - 5(A+B)}{2(A-B)} h$$

Proceeding with the left and right columns of the window in the same way, we obtain the value of b as follows:

$$b = \frac{S_R - S_L}{2(A-B)}$$

To estimate A and B we could use these expressions:

$$A = \frac{1}{3}(F_{i,j+2} + F_{i+1,j+2} + F_{i+1,j+1})$$

$$B = \frac{1}{3}(F_{i-1,j-1} + F_{i-1,j-2} + F_{i,j-2})$$

Once we know the coefficients of the straight line that represents the edge, we could obtain the sub-pixel position (a units in vertical from the center of the pixel), and the normal vector, given by the expression:

$$N = \frac{A-B}{\sqrt{1+b^2}} [b, -1]$$

The magnitud of this vector represents the exact change of intensity at both sides of the edge.

2.1. Second order edges

If we approximate the edge as a second order curve, $y = a + bx + cx^2$, we could remake the expresions of the previous subsection. In this case, a system of 3 equations will be obtained using the sums of the 3 columns of the window. Then, the values of coefficientes a, b, c are obtained solving the system. Using these values, the edge features (sub-pixel position, direction and curvature) could be computed with higher precision that in the linear case.

3 Edge detection in smoothed images

The adquisition process adds noise to the image, and this noise produces small errors in the image values. The traditional way to diminish this error is convolving the image with a smooth mask, like a gaussian mask. The simplest mask is a $3x3$ mask like the following:

$$H = \begin{pmatrix} a_{11} & a_{01} & a_{11} \\ a_{01} & a_{00} & a_{01} \\ a_{11} & a_{01} & a_{11} \end{pmatrix}$$

where $a_{00} > a_{01} > a_{11} > 0$ and $a_{00} + 4a_{01} + 4a_{11} = 1$.

Let us be G the result image of convolving the original image F with the smooth mask H. We are interested in obtain the edge features in image F but using only the values of G.

Let us be $P_{i,s,t}$ the area inside the column i of the image F that is under the edge line. This column is composed of the pixels (i, s), $(i, s+1)$, ...,(i, t). If we assume that the edge line cross the column from left to right, its expression will be as follows:

$$P_{i,s,t} = \int_{(i-\frac{1}{2})h}^{(i+\frac{1}{2})h} \left(a + bx + cx^2 + \left(t + \frac{1}{2}\right) h\right) dx$$

Let us be $M_{i,s,t}$ the sum of all the pixel values of that column in the image F. Its expression will be:

$$M_{i,s,t} = \sum_{j=s}^{t} F_{i,j} = \frac{1}{h^2}(A-B)P_{i,s,t} + B\,(t - s + 1)$$

Finally, let us be $S_{i,s,t}$ the sum of the same pixels, but using the smoothed image G. Due to the convolution, the value of each pixel of G depends of a $3x3$ neighboorhood of that pixel in F, and therefore the expression of S will depend of different M. In this way, a three equations system could be proposed using the sum of the 3 columns of a window centered on the pixel (i, j), called from left to right S_L, S_M and S_R, as follows:

$$\begin{cases} S_L = S_{i-1,j+l_1,j+l_2} \\ S_M = S_{i,j+m_1,j+m_2} \\ S_R = S_{i+1,j+r_1,j+r_2} \end{cases}$$

where $l_1, m_1, r_1 < 0 < l_2, m_2, r_2$ are those that the absolute value of partial derivatives in the pixels $(i-1, j+l_1)$, $(i-1, j+l_2)$, $(i, j+m_1)$, $(i, j+m_2)$, $(i+1, j+r_1)$ and $(i+1, j+r_2)$ of the image G are miminum in each column. The coefficients of the curve are obtained solving the system, and edge features in the original image F are computed starting from them.

Figure 2 shows the result of applying this method to an ideal synthetic circle with noise added. To visualize

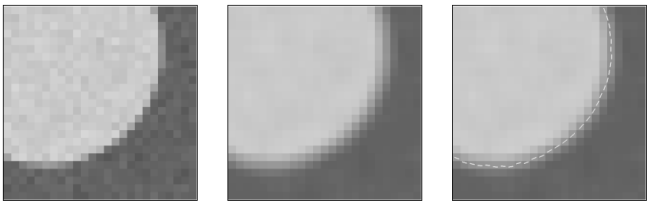


Figure 2: From left to right: noisy image, restored image, detected edges using the restored image

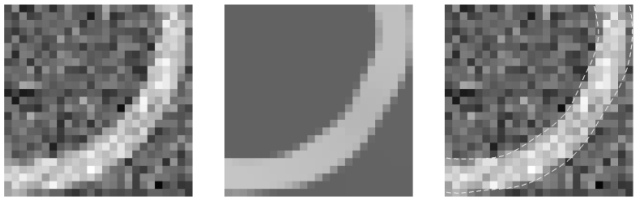


Figure 3: From left to right: noisy image, restored image, detected edges using the restored image

the result, a zoomed image has been generated, and inside each edge pixel a small straight line is drawn. This line indicates the direction and sub-pixel position estimated for that pixel.

4. Image restoration

When noise is too high, the usual method consist of defining an iterative process that modifies the image gradually, removing noise in each iteration. In this way, at the end of the process, an image with better quality is obtained. Finally, edge features are computed using the values of the restored image.

The algorithm we propose in this paper is the following: first, the original image F_0 is smoothed using a $3x3$ smooth mask, obtaining the smoothed image G_0. Secondly, the edge detector of the previous section is applied to this image. This methods will estimate the edge features at every edge pixel using a different neighboorhood for each one.

In those pixels where an edge is detected, a synthetic subimage could be generated that contains an ideal edge, using the values of the obtained features (sub-pixel position, direction, curvature and change in intensity at both sides). In this way, we could generate a complete synthetic image F_1, combining all these subimages. In those pixels where there is no neighboring edge, we keep the same value that in G_0.

The image F_1 will have better quality than F_0, and therefore edge features computed with it will be more

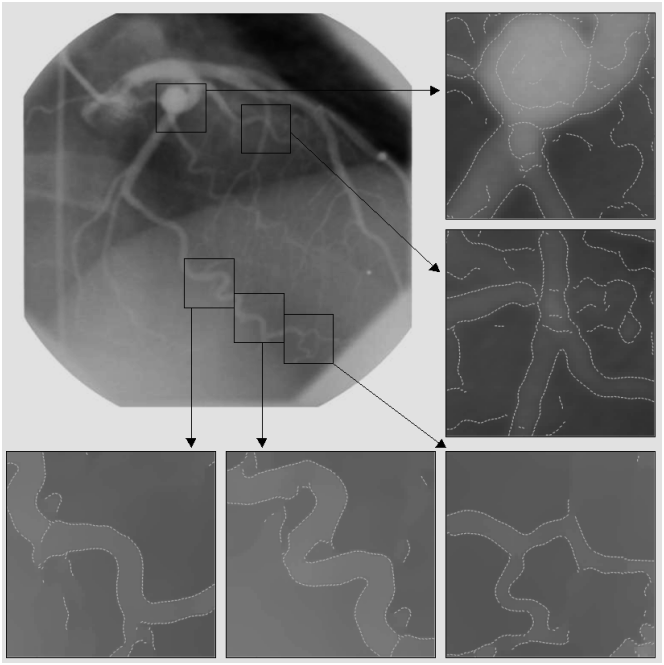


Figure 4: Detected edges on an angiography

precise. In fact, this scheme could be applied n times to obtain an image F_n with much better quality. Applying the proposed edge detector to the smoothed image G_n we obtain much better estimations of these features.

Figure 3 shows the result of applying this scheme to a synthetic image with noise added. Figure 4 shows the result to a real angiographic image.

5. Summary and Conclusions

We have presented an edge model that does not assume continuity in the image values, and an adquisition model based on regions of different intensities inside each edge pixel. Next, using these models, we have developed a method to estimate with high precision edge features as sub-pixel position, direction, curvature and change in intensity at both sides of the edge. Finally, using this detector, an iterative scheme has been proposed to improve image quality, in order to estimate features with higher precision in noisy images.

References

[1] D.Braggins, "Achieving Subpixel Precision", *Sensor Review*, 10(4), pp.174-177, 1990.

[2] K.R.Castleman, *Digital Image Processing*, Prentice-Hall, 1995.

[3] T.Lindeberg, *Scale Space Therory in Computer Vision*, Kluwer Academic Publishers, 1994.

[4] D.K.Naidu, R.B.Fisher, "A Comparative Analysis of Algorithms for Determining the Peak Position of a Stripe to SubPixel Accuracy", *Proc 1991 British Machine Vision Association Conf.*, pp.217-225, 1991.

[5] P.Perona, J.Malik, "Scale Space and Edge Detection using Anisotropic Diffusion," *IEEE Trans Pattern Anal. Mach. Intell.*, Vol 12, pp. 629-639, 1990.

[6] A.Rosenfeld, A.C.Kak, *Digital Picture Processing,* Academic Press, NY, 2nd edition, 1982.

[7] M.Sonka, V.Hlavac, R.Boyle, *Image Processing, Analysis, and Machine Vision*, PWS Publishing, 1999.

[8] A.Trujillo, "SubPixel Location of Edges based on an Invariant Gradient Vector Field Estimation," *IX Spanish Symposium on Pattern Recognition and Image Processing 2001.*

Recovering Human Shape and Motion from Video Sequences

Pascal Fua
Swiss Federal Institute of Technology
Switzerland
pascal.fua@epfl.ch

Abstract

In recent years, because cameras have become inexpensive and ever more prevalent, there has been increasing interest in modeling human shape and motion from image data. Such an ability has many applications, such as electronic publishing, entertainment, sports medicine and athletic training. This, however, is an inherently difficult task, both because the body is very complex and because the data that can be extracted from images is often incomplete, noisy and ambiguous.

In this talk, I will present the approach we have developed to overcome these difficulties. We start from sophisticated 3-D animation models and reformulate them so that they can be used for data analysis. We use them, not only to represent faces and bodies in motion, but also to guide the interpretation of the image data, thereby substantially improving performance. Using complex video sequences, I will highlight the effectiveness of our approach to video-based shape and motion capture and demonstrate the applicability of our technology for Augmented Reality purposes. Finally, I will present some open research issues and discuss our plans for future developments.

.

Tracking facial points from calibrated stereo

N. Pérez de la Blanca[1], J. M. Fuertes[2], M. Lucena[2]

[1]Department of Computer Science and Artificial Intelligence
ETSII. University of Granada, 18071 Granada, Spain
nicolas@ugr.es
[2]Departmento de Informática. Escuela Politécnica Superior. Universidad de Jaén
Avenida de Madrid 35, 23071 Jaén .Spain
{jmf, mlucena}@ujaen.es

Abstract. This paper proposes an approach to track facial landmark points through a color stereo image sequence. We assume a calibrated and rectified stereo pair. The approach only uses the color information of the images; neither new hypothesis nor graphic models of the face are required. The approach uses the disparity space representation as the main space where carrying out all fitting process. A robust iterative linear algorithm based on the algorithm RANSAC is used to estimate the rigid motions of the face through the image sequence. This approach is based on the fact that the disparity information is essentially robust against facial local motions. So we transform the stereo image sequence in a disparity space sequence, and we estimate the rigid motion between each two consecutives images from a homography between the corresponding disparity spaces. Here, the dense disparity map is used as a feature from which to estimate the homography parameters. After compensating the rigid motion effect, an accurate estimation of the local motion is given from a local matching process on the color information of the image. The approach has several potential advantages on the use of mesh graphical model. Firstly, no graphical template is used to track the landmark points, and as a consequence we do not have the initialization problem; secondly, the set of points we want to track is not fixed beforehand as happen in a mesh graphical model. In our case any point can be tracked once the corresponding homography is estimated.

1 Introduction

Characterization of facial expressions from geometry in monocular image sequences has been one of the most active topics in the computer vision community [15]. The increase in the importance of applications based on the facial motion has developed different approaches, to this problem, depending on the application requirement [1-4] [6][12-14][15][19]. The vast majority of these approaches are based on the tracking of a set of facial landmark points through the image sequence [2][9][11]. In order to track the set of landmark points, mesh graphical models mixing different types of information, extracted from the facial surface, have been proposed [6][9][10]. The local motion estimation of each landmark point, from one image to the next, is given by the Euclidean distance between the image coordinates of the landmark point and the image coordinate of its corresponding nodal point in the template. There are, however, two important shortcomings in this approach. One is that an adequate graphical template must be available before starting the tracking process. The other is that the rigid motion of the head has to be estimated from the motion of the template nodes that we assume are located on rigid patches of the face. Unfortunately, the skin of the human face deforms in a very free way and it is difficult to define templates including nodal points that can be considered static in general. Therefore, alternatives to the traditional use of mesh graphical models must be considered. Other associated inherent difficulties to the graphical templates use are: i) estimation of the initial position of the template on the first image of the sequence, and ii) automatic adaptation of the template to a new facial pose when not all the landmark points are visible.

In this paper, we propose an approach based on stereo images that have several potential advantages on the use of mesh graphical model. Firstly, no graphical template is used to track the landmark points, and as a consequence we do not have the initialization problem; secondly, the set of points we could track is not fixed beforehand as happen in a mesh graphical model. In our approach any point can be tracked once the corresponding homography is estimated.

Our approach is based on the fact that the disparity information is essentially robust against facial local motions. So we transform the stereo image sequence in a disparity space sequence, and we estimate the rigid motion between each two consecutives images from a homography between the corresponding disparity spaces [7][8]. Here, the dense disparity map is used as a feature from which to estimate the homography parameters.

Since we are interested in studying 3D object motions close to the camera, we use the general perspective camera model in order to analyze our images. An important instance of this situation appears in 3D videoconferencing systems, where the 3D shape of

the head and face of each participant must be refreshed in each instant of time, and the usual short distance between cameras and surfaces introduces strong perspective effects. In the case of monocular images, affine deformations have been successfully approached [1] [3][4], but the case of perspective deformation in our knowledge remains open.

In Section 2, we introduce the geometrical concepts of the stereo images. In Section 3 the tracking algorithm is presented. In Section 4, the disparity map estimation is discussed. In Section 5, experiments carried out and the results are shown. Summary and conclusions are presented in section 6.

2 Stereo Images

Let us consider a calibrated rectified stereo rig, *i.e.* the epipolar lines are parallel to the ***x***-axis. There is no loss of generality since it is possible to rectify the images of a stereo rig once the epipolar geometry is known [11]. We also assume that both cameras of the rectified stereo rig have internal parameters that are similar and known.

Let us consider a rectified image pair, and let (x,y) and (x',y') be the two corresponding points in that image pair. Since the corresponding points must lie on the epipolar line, the relation between the two points is $x'=x-d$, $y'=y$ where d is defined as the disparity of the point (x,y). From rectified stereo images, we can define *disparity spaces* that are equivalent to a 3D reconstruction of the points up to a homography of the 3D space [8]. The equations relating the coordinate vector (X,Y,Z) of a 3D point to its disparity coordinate vector $(\bar{x}, \bar{y}, d)^T$ in the case of oriented and rectified cameras are [19]:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \frac{B}{\bar{x}-\bar{x}'} \begin{pmatrix} \bar{x} \\ \bar{y} \\ 1 \end{pmatrix} \qquad \bar{x}=\frac{x-x_0}{\alpha}, \quad \bar{y}=\frac{y-y_0}{\alpha}, \quad \bar{x}'=\frac{x'-x'_0}{\alpha'} \tag{1}$$

where x_0, y_0, x'_0 are the principal point coordinates of the left and right image, respectively, α and α' are the focal distance of the left and right cameras, respectively and B is the baseline of the stereo rig. All image coordinates are expressed in terms of pixels. From expression (1), assuming $\alpha=\alpha'$, the homographic relationship between the 3D coordinates of a point $\mathbf{X}=(X,Y,Z)^T$ and its associated disparity vector $(\bar{x}, \bar{y}, d)^T$ can be expressed as

$$Z\begin{bmatrix} \bar{x} \\ \bar{y} \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & 0 & 0 \\ 0 & \alpha & 0 & 0 \\ 0 & 0 & 0 & \alpha B \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3}$$

or in a shorter way as

$$\begin{pmatrix} \mathbf{t} \\ 1 \end{pmatrix} \cong \mathbf{?}_B \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \qquad , \quad \mathrm{t}=(\bar{x}, \bar{y}, d)^T \tag{4}$$

2.1 Rigid motions in the disparity space

Let us apply a rigid motion on the 3D data. If $\mathbf{X}$ and $\mathbf{X}'$ represent the 3D coordinates of a point before and after the motion, then

$$\begin{pmatrix} \mathbf{X}' \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \tag{5}$$

From expressions (4) and (5) we obtain

$$\lambda \begin{pmatrix} \tau' \\ 1 \end{pmatrix} = \mathbf{H}_B \begin{pmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix} \mathbf{H}_B^{-1} \begin{pmatrix} \tau \\ 1 \end{pmatrix} = \mathbf{G} \begin{pmatrix} \tau \\ 1 \end{pmatrix} \tag{6}$$

Equation (6) describes the 3D homography G relating the disparity homogeneous coordinates of a point before and after the motion .

3 Tracking Algorithm

3.1 Noise on the data

The source of the noise in the disparity space is known. The disparity coordinates $\bar{x}$ and $\bar{y}$ are affected by the noise produced by the discretization effect and without additional information can be assumed equal for all pixels. The noise on *d* is associated to the change in the gray level of the pixels in the stereo matching process and could be estimated from this process. We can therefore assume that the associated noise to $\bar{x}, \bar{y}$ and *d* are independent. Although there is no reason to consider the variance of *d* equal to the variance of the discretization error, the covariance matrix of the noise on each point of our disparity space can be taking as $W=\sigma^2 \mathbf{I}_{3x3}$ in case we do not have additional information. Other important noise source, in our case, is the local motion of the points. All the correspondences between points affected of local motion are erroneous according to the correspondences defined by the 3D rigid motion. So, in order to estimate the rigid motion, a robust algorithm selecting correspondences free of this noise must be considered. An important consequence of the local motion noise is the difficulty of using set of interest points in our estimation algorithm, since we do not know in advance if these points are affected or not by this contamination.

3.2 Rigid motion estimation

Let (t_i, t'_i) be a set of point correspondences. The problem of estimating the rigid motion parameters (**R,T**) from the set of points (t_i, t'_i) amounts to minimizing the error

$$E^2 = \sum_i d(\tau'_i, \mathbf{G}\tau_i)^2 , \quad d(\tau'_i, \mathbf{G}\tau_i)^2 = (\tau'_i - \tau'^{\Gamma}_i)^T \mathbf{O}^{-1} (\tau'_i - \tau'^{\Gamma}_i) \qquad (7)$$

where $\tau'^{\Gamma}_i = (\tau'^{\Gamma}_{i1}/\tau'^{\Gamma}_{i4} \quad \tau'^{\Gamma}_{i2}/\tau'^{\Gamma}_{i4} \quad \tau'^{\Gamma}_{i3}/\tau'^{\Gamma}_{i4})$ is the estimated Euclidean coordinate vector for τ'_i from (6), and W is the covariance matrix of the disparity vectors [7]. Here we assume an i.i.d noise model. Equation (6) shows that this error function is not linear in the parameters for (**R,T**), so an cuasi-linear iterative algorithm has been used to estimate the vector of six unknowns by parameterizing the rigid motion. Here we are interested in small rotations (< 5 degree), so the rotation matrix can be expressed as $\mathbf{R}=\mathbf{I}+[w]_x$, where I is the identity matrix and $[w]_x$ represents the skew-symmetric matrix associated to the vector *w*. An initial solution for the vector $(w,T)^T$ can be calculated from equation (6), solving the linear system that appears by considering the equations associated to Euclidean coordinates of all the points τ and τ' and assuming all λ=1. In the next iteration we recalculate the value of λ from the above solution and again solve equation (6) for a new solution. We iterate until convergence of the vector $(w,T)^T$. In our experience, three or four iterations are sufficient
Nevertheless, the presence of outliers in the correspondences between the disparity maps degrades the estimation considerably. In order to circumvent this problem a RANSAC based algorithm is proposed in Table 1.

```
Iterative robust algorithm:
I. To estimate and normalize the set of disparity vectors
II. To cluster the disparity vectors in classes according to its
disparity value.
III. Repeat N iterations
     i. To random n>2 classes. To take one vector from each class.
            For each vector to calculate li, Ai and bi
     ii. Solve the full system lAX=b for X
     iii Count the number of inliers on all the correspondences.
IV. To take as solution that with the highest number of inliers.
```

Table 1.

We cluster the disparity vector in classes in order to benefit samples with different disparity points. The regions with higher disparity have an definitive influence in the accuracy of the rigid motion estimation, but these regions in general are smaller in number of pixels.

3.3 Local matching

The estimated homography, between each two consecutive disparity spaces, compensates the rigid motion of the head from one image to the other. But in order to estimate the local motions of the face we carried out a matching step by local search. We start transferring, by the homography, the image coordinates of the points from the first image to the second. Then a crosschecking cross correlation matching on a 17x17 size search window around each transferred coordinated is carried out. The 13x13 size correlation window is adapted in each point from the rectified color values defined by the homography. Although, the searching pattern defined by the correlation window it is not the real one, it can be assumed that it is very close to it. In local patches with low texture, however, this fitting step can produce small oscillations around the true location. As figure of merit to minimize we use the normalized cross correlation.

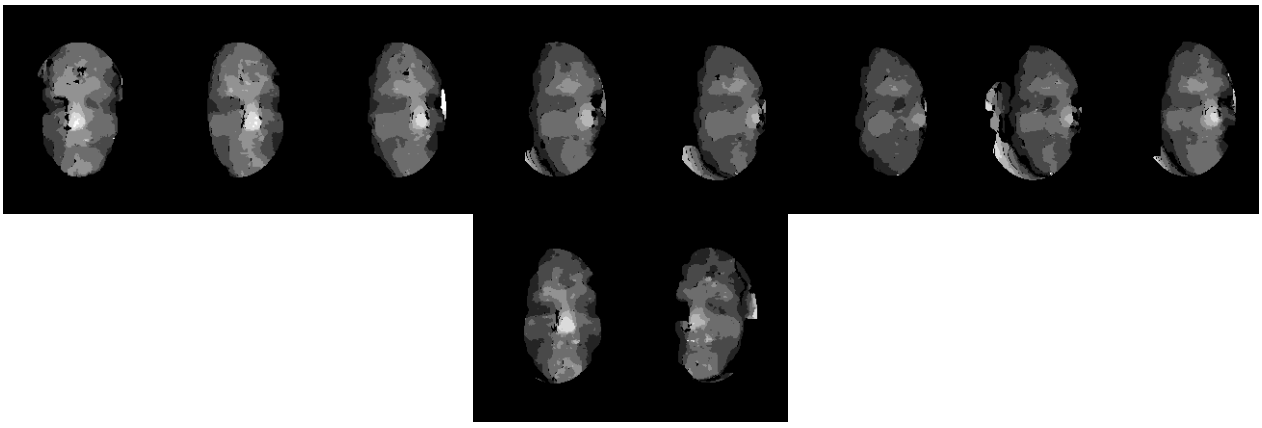
4 Disparity map estimation

We estimate two dense motion maps in this approach. Firstly, we estimate the disparity map for each stereo image, and secondly, we estimate the dense motion vector map associated with every two consecutive left and right images, respectively. From the disparity map we segment a region of interest by applying a binary threshold on the disparity value range. The threshold is fixed to define our region of interest. From the motion vector map we segment a new region of interest defined by the pixels where the motion vector norm is higher than a fixed threshold. The intersection of both regions segments the subset of moving points of the scene with disparity value higher than the fixed threshold. In order to remove isolated small regions we apply a size filter to the intersection region. Figure 1 shows an example of the region of interest estimation process.

(a) (b) (c) (d)

Fig. 1. This example corresponds to rotation left-right of the head. Picture (a) represents the estimated stereo disparity map, picture (b) represents the x-motion dense map, picture (c) represents the y-motion dense map, and picture (d) represents the result of the union of picture (b) and picture (c) intersection with picture (a).

Dense disparity maps estimation from two images is a very active field of research with many different techniques classified as local o global approaches [5][16]. We have combined a local approach to obtain the maximum possible accuracy with a smoothing surface-fitting step to fill the holes. We start applying a crosschecking adaptive multiple window algorithm, minimizing the normalized correlation, on a fixed number of scales. We calculate the disparity of each pixel in all the scales and we take out that with higher correlation value. Once we get this first disparity raw estimation, we fit a smooth surface using a penalized conjugate gradient algorithm [17]. We also have experimented with global methods based on cut graphs algorithms [16]. In our experience, however, although these methods always return a dense disparity map, the estimation is too smooth to be useful in the rigid motion estimation. In Fig.2 we show some of the obtained disparity map estimations.



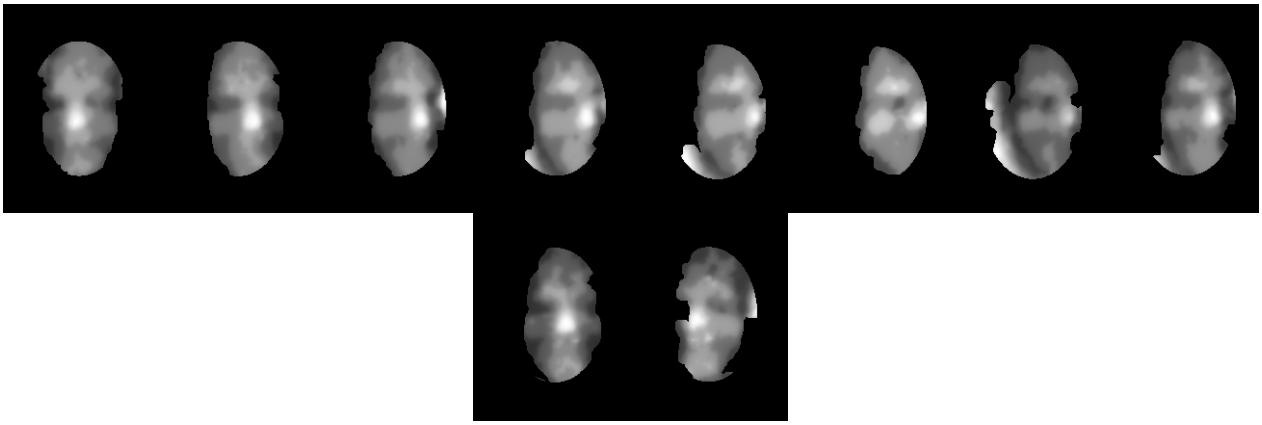


Fig. 2. This figure, from left to right, shows samples of the estimated disparity map sequence for the image sequence shown in figure.3. The first two rows show the raw disparity maps. The second two rows show the result of fitting a surface to the above raw estimations. All these images correspond to the left image of the stereo pair. The higher gray level the higher disparity.

From Figure 2 we can evaluate the quality of the estimation obtained. It can be appreciated, as the disparity maps are coherent with the 3D rigid motion of different parts of the face. Some estimations present noisy patches that could influence the rigid motion estimation. But, in our experience the algorithm RANSAC has managed this situation very properly. The combination of estimations obtained from the different scales has mainly beneficed the disparity estimation of the textureless region of the face.

5 Experimental results

Experiments to estimate 3D facial motions have been carried out from different 640x480 stereo image sequences captured by a factory calibrated Pointgrey stereo camera (Bumblebee) with 4mm focal length lens, watching an actor moving his face freely to a distance to the camera lower than one meter. A fixed window inside the captured images fixed the sub-images of interest. Our algorithm was applied to the image sequence defined by the sub-images. The proposed algorithm was applied on every two consecutive stereo images in the sequence. In order to assess the quality of the rigid motion estimation process we synthesized a new sequence of images by interpolating from the estimated motions and the original sequence. In the cross-correlation disparity calculation we use the scales defined by odd size correlation windows ranging from 7 to 15.

Figure 3 shows six equally spaced samples taken from an image sequence of fifty images. It can be seen the strength of the rigid and local facial motions respectively. Figure 4 shows how accurate the estimated rigid motion for this particular sequence is. We compare the norm of the difference between two consecutive images, with the norm of the residual calculated by the difference between an original image and its corresponding synthetic. The large decrease in the norm of the difference image from the first case to the second case, shows that the rigid motion is compensated enough.

Fig. 3. The first and second rows show a summary of the left and right images respectively of a stereo sequence of fifty images. Strong rigid and local motions can be appreciated along the image sequence.

The first row of figure 5 shows the result of applying the proposed algorithm on the stereo image sequence shown in figure.3. The landmark points, in white, were selected by hand in the first image of the sequence and the proposed algorithm automatically calculated their locations in the rest of the images. It is shown, as the locations of the landmark points are stable independently of the facial changes, although some of the selected points present difficulty in tracking. In particular, the point just on the tip of the nose is a difficult point since the disparity of this point is usually underestimated in the disparity fitting process. As a consequence, higher transfer error is obtained by the homography. There are, however, unsolved difficulties in our approach. The most important is the recovery of the landmark point location after an occlusion. This situation is present in figure 5 from the third sample. It can be seen how the right extreme of the right eyebrow disappear as consequence of the head rotation. In this case the fitting process follows the texture of the eyebrow but eventually loss the point, as it can be appreciate in the last picture. It also can be observed how some of the points oscillate around its true location. This is a consequence of the use of cross-correlation and the texture variation with the motion. A possible improvement to the use of gray values cross-correlation, in this second step, would be the use of illumination and motion invariant vectors characterizing the matching area.

.

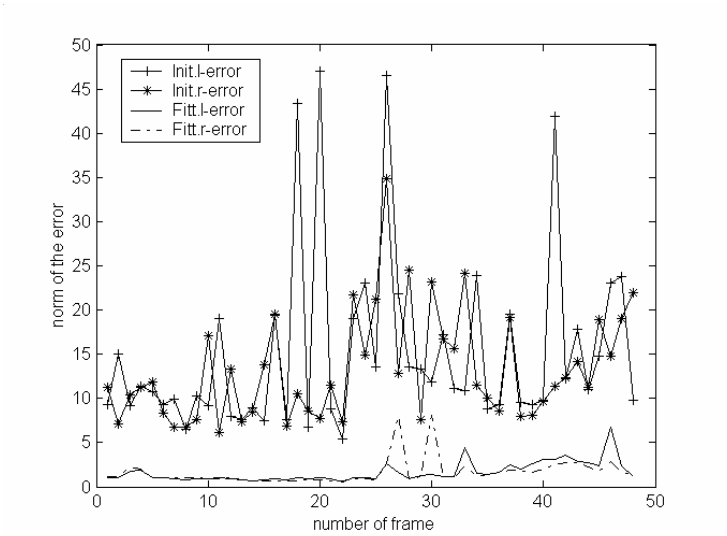


Fig. 4. This figure shows four graphs each of which is the norm of the gray level difference pixel-by-pixel from two images. The graphs Init.l-error and Init.r-error show when the images are two original consecutive right images and left images, respectively. The graphs Fitt_l.error and Fitt_r.error show when the two images are the original and synthesized, by the estimated homographies, for the right and left images, respectively.

The proposed technique is time efficient since the costliest step is the disparity estimation by cross-correlation which can be implemented efficiently. In the algorithm RANSAC step of the homography estimation process a maximum of 300 samples of three elements each were sampled.

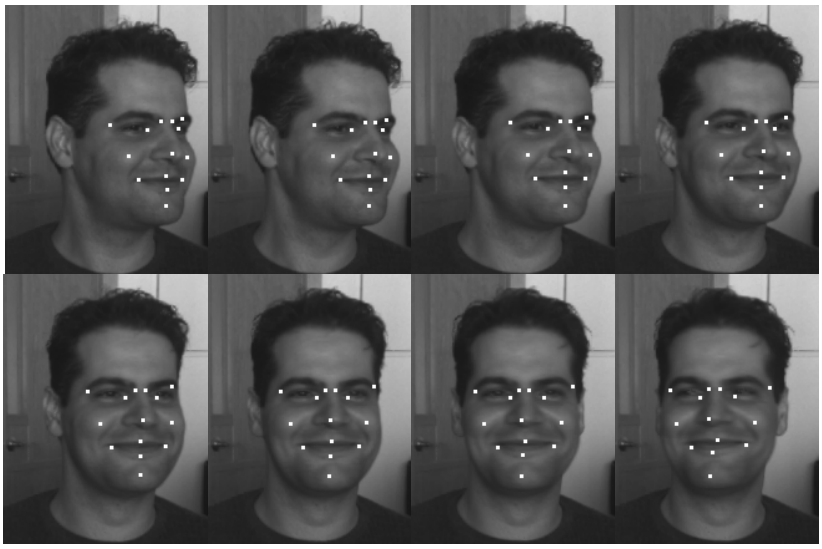


Fig. 5. This figure shows, from left to right, a subsequence of images from the original sequence. In white, shows the estimated location of the selected set of landmark. The first image (top-left) shows the initial locations. All images belong to the left camera of the stereo pair

6 Summary and Conclusions

A new approach to the moving and static facial point-tracking problem from stereo geometrical information has been given. A robust algorithm to estimate the rigid motion of the head between consecutive images has been proposed. We have shown that the dense disparity map is a robust and useful feature in order to estimate the rigid motion parameters. The use of a homography to transfer static and moving points between images, it is a clear improvement in comparison to the approaches to the same problem in monocular sequences, where a similar transfer mechanism for points in motion does not exist. In this approach only the observed image data and the camera calibration data has been used in order to regularize all the fitting process. Furthermore, the set of points to track does not need to be fixed in advance as happen when a graphical templates is used. Any point in the region of interest can be tracked once the homograhies are estimated. But in order to have a definitive approach the recovery of the landmark location after occlusion must be solved. In the same way, more sound and elaborated window matching mechanisms, invariants to 3D rigid and local motions, have to be approached in order to increase the accuracy of the local matching step.

Acknowledgments

This work has been financed by the grant TIC2001-3316 from the Spanish Ministry of Science and Technology.

References

1. Bascle, B., and Blake, A.: Separability of pose and expression in facial tracking and animation. In Proc. Int. Conf. Computer Vision. 1998.
2. Basu, S., Oliver, N., Pentland A.: 3D modelling and tracking of human lip motions, ICCV'98, 1998.
3. Brand, M. : 3D Morphable models from video. In Proc. CVPR'01. 2001
4. Bregler, C., Hertzmann, A., Biermann, H.: Recovering non-rigid 3D shape from image streams. In Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'00) . 2000.
5. Brown, M.Z., Burschka,D., and Hager,G.D.: Advances in computational stereo, IEEE Transc. on PAMI, vol-25,8,993-1008,2003.
6. Decarlo, D., and Metaxas, D.: Optical flow constraints on deformable models with application to face tracking. International Journal of Computer Vision, 38(2),99-127,2000.
7. Demirdjian, D., and Darell, T.,: Motion estimation from disparity images, In Proc. ICCV01, Vancouver Canada,vol-II, 628-635,2001
8. Devernay, F. and Faugeras, O.: From projective to Euclidean reconstruction. In Proceedings Computer Vision and Pattern Recognition,264-269,1996.

9. Fua, P.: Regularized bundle-adjustment to models heads from image sequences without calibration data, International Journal of Computer Vision, 38(2), 2000.
10. Fua,P., and Miccio C.: From regular images to animate heads: A least square approach, In Proceedings ECCV'98, Lecture Notes in Computer Science , 188-202,1998.
11. Hartley, R. and Zisserman, A.,: Multiple View geometry in computer vision. CUP, 2002
12. Huang,Y & Huang T.: Facial tracking with head pose estimation in stereo vision, Intern. Conference on Image Processing, 2002.
13. Lanitis, A., Taylor, C.J., Cootes, T.F. and Ahmed, T.: Automatic interpretation of human faces and hand gestures using flexible models. In International Workshop on Automatic Face-and-Gesture Recognition, 1995.
14. Pandzic,I.S. & Forchheimer,R. (eds): MPEG-4 Facial Animation, Wiley,2002.
15. Pighin,F, Szeliski,R., and Salesin,D.H.: Modeling and animating realistic faces from images, Intern. Journal on Computer Vision, 50(2),143-169,2002.
16. Scharstein, D. and Szeliski, R.: A Taxonomy and evaluation of dense two-frame stereo correspondence algorithms, Intern. Journal of Computer Vision, 47(1),76-81,1999.
17. Szeliski , R: Fast surface interpolation using hierarchical basis functions, IEEE Trans. On PAMI, vol-12,6,513-528,1999.
18. Tarel, J.P.: Global 3D Planar Reconstruction with Uncalibrated Cameras, A Rectified Stereo Geometry, Machine Graphics & Vision Journal, vol-6, 4, 393-418,1997
19. Valente.S. and Dugelay, J.L.: A visual analysis/synthesis feedback loop for accurate face tracking, Signal Processing Image Communications, 16, 585-608,2001

Face and Hands Segmentation in Color Images and Initial Matching

Jose Maria Buades Rubio Manuel González Hidalgo Francisco Jose Perales
Unidad de Graficos y Vision. http://dmi.uib.es/research/GV
Departamento de Matemáticas e Informatica
Universitat de les Illes Balears
Crta Valldemossa Km 7.5
Spain E-07122, Palma de Mallorca, Baleares
{josemaria.buades, dmimgh0, paco.perales}@uib.es

Abstract

In this paper we describe a new method for detection and initial pose estimation of a person in a human computer interaction in indoor uncontrolled environment. We used the Koepfler-Morel-Solimini mathematical formulation of Mumford-Shah segmentation functional adapted to color images captured from two cameras. The idea is to obtain a system to detect the hands and face in a sequence of binocular images. The skin color is predefined and a procedure is parameterized to segment and recognized the homogeneous regions. Also we fit our results to a restriction that the two hands and face must be detected at the same time. Also we used a biomechanical restriction to reach this initial estimation. So the centroid of the blob is computed for every region. The process is carried oout from two calibrated cameras that allow the system to reach a 3D reconstruction.

1 Introduction

In actual computers systems the interaction is going to a non-contact devices. That's means that allow the user to interact without physical contact with the machine; this communication can be carried out with voice or user gesticulation capture. We are especially interested in visual information, so recognize the human presence in colour video images. Also we would like to define a general, robust and efficient system that can be used with non-expensive cameras and digitalizing cards. Capture is carried out from colour cameras. The process is over 2D images, also we have done experiments with two calibrated cameras and a 3D reconstruction module build a 3D scene in real time.

The global process must detect a new user entering the system and analyse him/her to determine parameters such as hair colour and clothes. Once, the user who is going to interact with the machine has been detected, the system starts to track interesting regions such as the head, hands, body and joints, using information obtained in the user detection task. The input data for the gesture interpretation process are the position and orientation of these regions. This process will determine which gesture the user has carried out. Next, these gesture data are sent to the execution process, which ends the process by performing the action that has been specified, and so completing the feedback process.

In the following section, we explain briefly the mathematical background of the segmentation process. Section 3 introduces the main method to detect the user in front of the camera and carefully explains the analysis process and parameters needed for a future tracking process. Section 4 explains 3D reconstruction module from two calibrated colour cameras. Finally, we conclude with some new and extended results including a set of colour images and conclusions, future works and references. This work is a new version of [8] improving the segmentation process and the result ratio in classification process.

2 Multichannel Segmentation Algorithm

Image segmentation is the first step in data extraction for computer vision systems. Achieving good segmentation has turned out to be extremely difficult, and is a complex process. Moreover, it depends on the technique used to detect the uniformity of the characteristics sought between image pixels and to isolate regions of the image that have this uniformity. Multiple techniques have been developed to achieve this goal, such as contour detection, split and merging regions, histogram thresholding, clustering, etc. A Survey can be found in [1].

In color image processing, pixel color is usually determined by three values corresponding to R (red), G (green) and B (blue). The distinctive color sets [7] have been employed with different goals, and specific sets have even been designed to be used with specific segmentation techniques [1].

We define a color image as a scalar function $g = (g^1, g^2, g^3)$, defined over image domain $\Omega \subseteq \Re^2$ (normally a rectangle), in such a way that g: $\Omega \rightarrow \Re^3$. The image will be defined for three channels, under the hypothesis that they are good indicators of autosimilarity of regions. A segmentation of image g will be a partition of the rectangle in a finite number of regions; each one

corresponding to a region of the image where components of g are approximately constant. As we will try to explicitly compute the region boundaries and of course control both their regularity and localization, we will employ the principles established in [2, 4, 6] to define a good segmentation. To achieve our goals we consider the functional defined by Mumford-Shah in [3] (to segment gray level images) which is expressed as:

$$E(u,B)=\int_{\Omega}\sum_{i=1}^{3}(u^i-g^i)^2 d\mu+\lambda\ell(B) \quad (1)$$

where B is the set of boundaries of a homogenous region that define a segmentation and u (each u^k) is a mean value, or more generally a regularized version of g (of each g^k) in the interior of such areas. The scale parameter λ in the functional (1) can be interpreted as a measure of the amount of boundary contained in the final segmentation B: if λ is small, we allow for many boundaries in B, if λ is large we allow for few boundaries.

A segmentation B of a color image g will be a finite set of piecewise affine curves - that is, finite length curves - in such a way that for each set of curves B, we are going to consider the corresponding u to be completely defined because the value of each u^i coordinate over each connected component of $\Omega \setminus B$ is equal to the mean value of g^i in this connected component. Unless stated otherwise, we shall assume that only one u is associated with each B. Therefore, we shall write in this case $E(B)$ instead of $E(u, B)$. A segmentation concept which is easier to compute is defined as follows:

Definition 1. A segmentation B is called 2-normal if, for every pair of neighboring regions O_i y O_j , the new segmentation B' obtained by merging these regions satisfies E(B') > E(B).

We shall consider only segmentations where the number of regions is finite, in other words $\Omega \setminus B$ has a finite number of connected components and the regions do not have internal boundaries.

A more detailed explanation of the concepts and their mathematical properties can be consulted in [2, 4] and we can see the properties of the functional in [3,4]. The use of multichannel images (eg. color images) can be seen in [4, 5]. We shall use a variation of segmentation algorithm by region merging described in [2] adapted to color images.

The concept of 2-normal segmentations synthesizes the concept of optimal segmentation we are looking for, and it lays on the basis of the computational method we use. The 2-normality property is well adapted for the construction of an algorithm based on region growing by merging neighboring regions. Two regions will be merged if this operation reduces the energy. At each step we need to compare the balance of energy if we remove a common boundary $\partial(O_i, O_j)$ of two neighboring regions O_i, O_j. If B is 2-normal, one has $E(B) \leq E(B - \partial(O_i, O_j))$, which, in the case of a piecewise constant function u, implies the balance

$$\lambda\ell(\partial(O_i,O_j)) \leq \frac{|O_i|\cdot|O_j|}{|O_i|+|O_j|}\left(\sum_{k=1}^{3}(u_i^k-u_j^k)^2\right) \quad (2)$$

where $|\cdot|$ is the area measure and u_i, is the approximation of g on O_i to compute the data for evaluating the balance for each region O_i we associate its area $|O_i|$ and we can compute $u_i^k=\int_{O_i} g^k/|O_i|$ for k=1, 2, 3.

We call equation (2) the merging criterium. We decide to remove the common boundary $\partial(O_i, O_j)$ of O_i and O_j if this equation is not satisfied. By repeating this step, that is, by comparing the balance energy for deciding to join any two neighboring regions, we finally obtain a 2-normal segmentation for the scale parameter λ, a segmentation, i.e., where no further elimination improves the energy. Then, we have implemented a multiscalar algorithm and data structure similar to that used in [2] but adapted to color images and real time processing.

The algorithm used the RGB components, because the segmentations obtained are very accurate to our goal. But the system is able to use another color space or color descriptor as we can see in [1]. Moreover, if it is needed it can weigh the channels used in order to obtain the segmentation.

3 User Detection and Initial Pose

The image is captured and segmented with the algorithm explained in the previous section and is then analyzed to determine whether it is a user or not, as we can see below in a work related with this topic [6]. If a user has been detected, the system studies him and obtains some parameters that will be useful in the tracking and analysis process [7]. By applying this process directly to segmented images without using information from previous frames, the system is robust to background changing and variable illumination. The parameters obtained from the segmentation task are fixed in order to user interactions with upper torso (body, arms, hands and head). The system obtains the upper torso configuration: shirt, hair, hands and face. User detection process is waiting for a user located opposite the camera, with hands separated and at the same height that head, then it recognizes and later analyzes user configuration.

This module receives a segmentation of the captured image, analyzes every region and marks as skin region if its RGB medium value is in a characteristic color range of skin. To achieve more homogenous regions, neighboring skin regions are merged. This merging is carried out to avoid detecting a hand or the face in two neighboring regions, following the merging criteria:

$$\forall O_i, O_j / Neighbour(O_i, O_j) \wedge$$
$$Skin(O_i) \wedge Skin(O_j) \Rightarrow O_i \cup O_j \quad (3)$$

where $Neighbor(O_i, O_j)$ means that two regions are neighbors and $Skin(O_i)$ means that is a skin region.

After this, we obtain a skin region set, called β, where any pair of skin regions are separated.

For all ordered set of three regions included in β, we identify each one as face Z, left hand Y_1 and right hand Y_2, then we evaluate a criteria to determine whether this configuration is correct.

$$\underset{i,j,k}{Max}\{\varphi(O_i, O_j, O_k) : \forall O_i, O_j, O_k \in \beta\} \geq \alpha \quad (4)$$

where α is a threshold probability and we call φ the user detection function. In this function we take into account the following:

- Central region, face, must be the biggest.
- Lateral regions, hands, have a similar area.
- Face region area A(Z) must be between a minimum Z^- and a maximum Z^+
- Hands area A(Y_1) and A(Y_2) must be between a minimum Y^- and a maximum Y^+
- Vertical position Y_1 and Y_2 should be similar and nearest possible to Z

The user detection function returns a value between zero and one that measures the probability that a user has been detected. From all possible combinations of Z, Y_1 and Y_2 the one with the greatest value, greater than a reference minimum value α, is chosen as the best configuration.

In order to apply the above algorithm, we need to fix the following values: a color range of skin to detect hand and face regions, a threshold probability α to discriminate non expected initial positions. To avoid high differences of hands we include an area similarity criterion, a maximum size of hand area is also necessary. All these parameters are used in order to discriminate bad detections.

All threshold values are established in relation with camera to user distance and image resolution. This distance is predefined by initial application setup.

After a user has been detected, the same image is analyzed to determine hair and shirt color. Region proposed as hair, X is the upper neighboring region of Z if A(X) / A(Z) relation is greater than a threshold, hair is discarded and is considered that it is a bald user.

To analyze shirt, the following algorithm is applied. Initially, shirt region W is the greatest region whose upper boundary is included in the boundary of Z (see Figure 1). Afterwards, neighboring regions of W are joined until Z is connected with Y_1 and Y_2 through W. A candidate region T_i chosen at every step i to be joined to W is in relation with: color space distance between mean color of T and W, and distance in pixels from T to Y_1 and Y_2.

With this process, the system detects a user and obtains useful data for the tracking system.

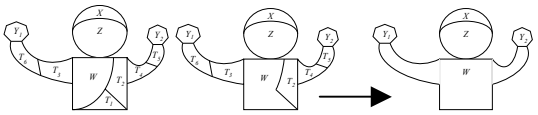


Figure 1. Shirt region detection

4 3D Reconstruction

To achieve a 3D reconstruction we merge data obtained from two calibrated cameras. We label camera 1 and 2. Every camera do separately the segmentation and user detection porcess. Calibration is done with a classical calibration algorithm without distorsion.

We use a non distorsion calibration because OpenGL doesn´t allow distorsion in its graphics pipeline, OpenGL is a graphics library developed to work in real time, others real-time graphics libraries don´t implement distorsion. To generate an approximated distored image is necessary to generate distorsion from the image generated by OpenGL, and the image quality and frame rate is decreased.

The 3D reconstruction receives as input data the regions detected obtanied from user detection. For every region: face, left hand and right hand we compute the center in the 2D image space from the two images. With regions center compute the 3D position intersecting the two lines that cross region center and camera position. This two lines practicaly never will intersect, then is computed as 3D position the point that is nearest to two lines.

At this point we have the 3D position for the face, right and left hand, and we need more information for obtain a 3D volume, the volume selected to model the regions is a sphere.

We have computed the center of the sphere, and the other parameter needed is the radius. First, the radius is computed in the image space as the circle that has the same areas that the region has been detected.

Second, the 3D radius is estimated for camera 1 and camera 2 taking into account the distance from the camera to the 3D position, and the radius of the region.

Last step merges 3D radius obtained from camera 1 and camera 2, 3D radius is the mean value for the two cameras.

Now, we build a 3D scene with 3D positions and 3D radius for left hand, face and right hand. The user navigate trought scene. All process is done in real-time, only a little delay is produced due to basically to segmentation process.

5 Results

We have implemented the above algorithm in C++. It has been tested in 320x240 resolutions (Figure 2) and 640x480 standard video resolution (Figure 3). We

initialize the multichannel segmentation algorithm with an initial segmentation wich is a grid of size T_x x T_y on the image, usually we take $T_x = T_y = 1$, 2 or 4. From this initial segmentation, the algorithm determines a 2-normal segmentation following the merging criterion described in (2) and the specifications of the algorithm described at the end of section 2. The stopping criterion can be: if the last level $\lambda = 2^n$ has been reached or if there is just one region left or if the desired number of regions is reached. In our displayed experiments the stopping criterion is to achieve a fixed number of regions. Then, we apply the algorithm described in section 3 where the selected parameters are detailed: Skin range color in HLS ([0-10], [20-230], [62-255])

In the two sequences of pictures we can see in Green the boundaries of hair region. The color Red is used for boundaries of hand and face regions, the centroid of these regions is visualized with a solid red square. In Pink we display the upper-torso boundary and finally we use Black and White for other regions detected for the segmentation algorithm.

In the first sequence we take a 2x2 initial segmentation and the system runs at 5 frames/second in a P4 1.6GHz. We display several different initial positions and cloth configuration; and we can see how the proposed method detects the interesting regions. In the second sequence, Figure 3, we display the same initial pose image and the results obtained with different size of initial segmentation, from top to bottom we use 1x1, 2x2, 4x4 respectively. Left column pictures are 320x240 and right ones are 640x480. In the first case, the system runs at 0.32 frames/second, 1.41 frames/sec and 6.70 frames/sec; and in the second 0.08 frames/second, 0.30 frames/sec and 0.82 frames/sec respectively.

Figure 4, we display from a population of 30 persons, where 29 persons have been perfectly detected. Figure 5 displays 3D reconstruction images generated by OpenGL.

6 Conclusions and Future Work

In this paper we have proposed a new system for user detecting for HCI that does not use background substraction, therefore the system is robust to environment and illumination changes. Moreover, it analyzes the user to determine parameters that will be useful for a future tracking process. The region segmentation process based on the Koepfler-Morel-Solimini algorithm adapted to multichannel images is sufficiently good and beneficial for our aims. Besides, the process is carried out in real time. The software implementation is efficient and OOP. The result of this process is the input of a tracking and reconstruction of an intelligent human computer interaction system. It remains as future work to do tracking of interesting body parts and to interpret movements in order to carry out action recognition that the user is performing. At the moment, we are working on particle filter tracking with a biomechanical model to reduce the search space solutions. Moreover, a stereo version is proposed to improve final results.

This paper is subsidized by the project IST-2001-32202 HUMODAN and CICYT TIC2001-0931.

7 Referecnes

[1] H.D. Cheng, X.H. Jiang, Y. Sun, JinGli Wang "Color Image Segmentation: Advances and Prospects", Journal of Pattern Recognition 34, (2001), pp. 2259-2281

[2] G. Koepfler, J.M. Morel, and S. Solimini, "Segmentation by minimizing a functional and the merging methods", SIAM J. on Numerical Analysis, Vol 31, No 1, Feb. 1994

[3] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and variational problems", Communications on Pure and Applied Mathematics, XLII(4), 1989

[4] J.M. Morel and S. Solimini. "Variational Methods for Image Segmentation", Birkhauser Verlag. 1995

[5] M. Gonzalez "Segmentación de imágenes en Color por método variacional". Proc. Del XIV C.E.D.Y.A. y IV C.M.A. pp 287-288, 1995.

[6] C. Ballester, V. Caselles and M. Gonzalez, "Affine invariant segmentation by variational methods", SIAM J. Appl. Math., Vol. 56, No 1, pp. 294-325, 1996

[7] G. Wyszecki and W.S. Stiles "Color Science: Concepts and Methods. Quantitative data and Formulae" Wiley, 2nd Edition, 1982

[8] I. Haritaoglu, "W4: Real-Time Surveillance of People and Their Activities" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 22 No8, pp 809-830, 2000

[9] H. Sidenbladh, M.J. Black and D.J. Fleet "Stochastic Tracking of 3D Human Figures Using 2D Image Motion" ECCV 2000.

[10] J.M. Buades, M. Gonzalez, F.J. Perales. "A New Method for Detection and Initial Pose Estimation based on Mumford-Shah Segmentation Functional". IbPRIA 2003. Port d'Andratx. Spain. June 2003. pp 117-125

Figure 2. Some results obtained in real time with a Sony VFW-V500 camera. Images are 320x240 resolution in RGB color.

Figure 3. Some results obtained with different size of initial segmentation.

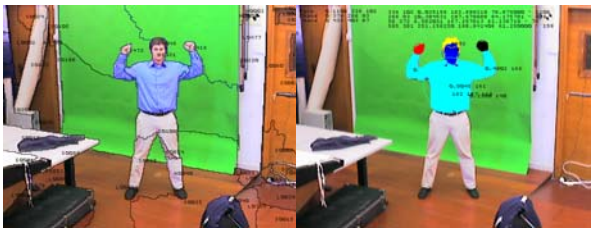
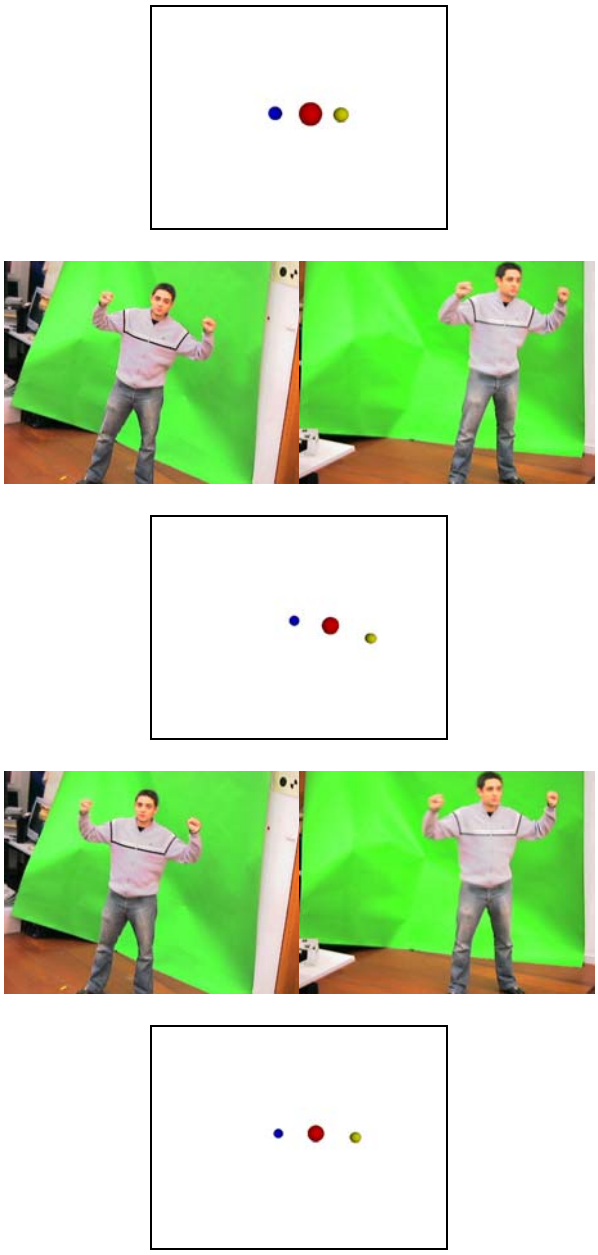


Figure 4. Some results from 30 persons test.



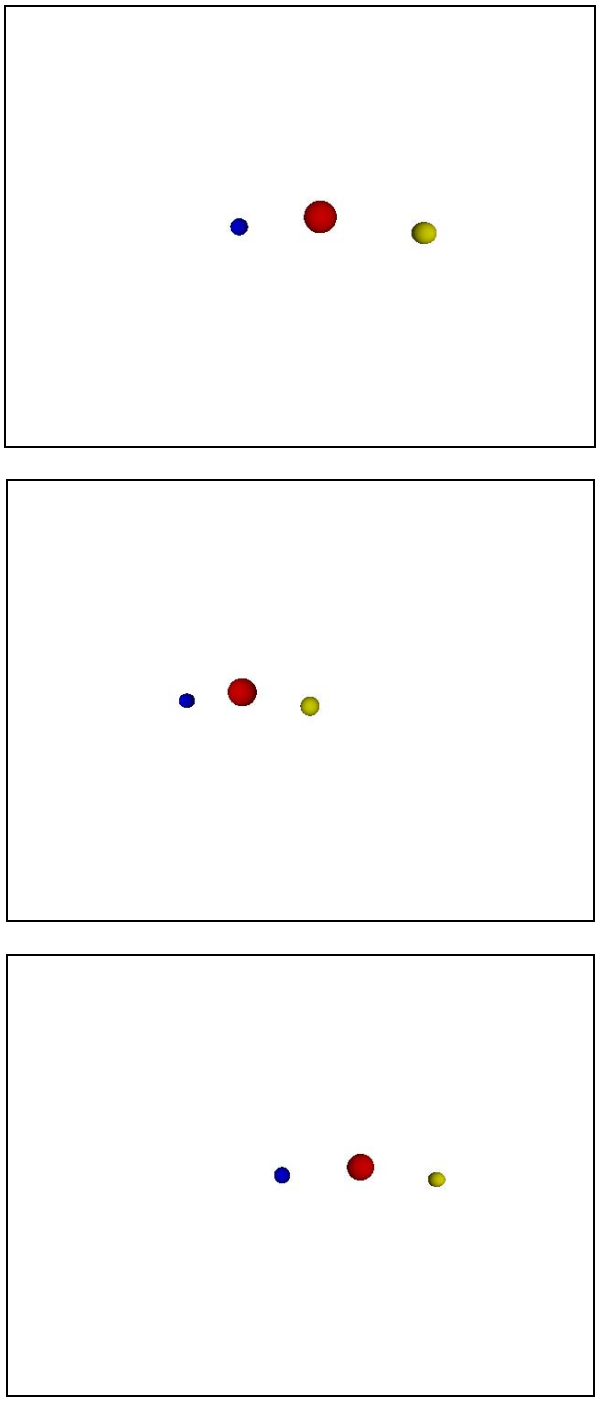
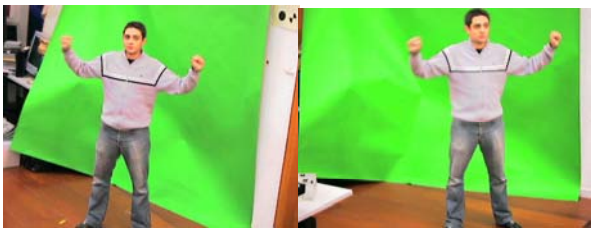


Figure 5. 3D Reconstruction. Captured images, frames 0, 50, 60 and 65, from camera 1 (left image) and camera 2 (right image), down OpenGL image build with 3D scene. For frame 65 (right column) three difernts points of view.



Real-time robust facial expressions analysis

José Miguel Buenaposada, Enrique Muñoz, Luis Baumela
Departamento de Inteligencia Artificial
Facultad de Informática, Univ. Politécnica de Madrid
Campus de Montegancedo s/n
28660 Boadilla del Monte, Madrid, Spain
Email: kike@dia.fi.upm.es, lbaumela@fi.upm.es

Abstract

In this paper we present a real-time face tracking system. It performs robustly under reasonable environment conditions (e.g. those in an office) and with a minimal initialisation: a skin colour patch, and an image sequence with the user performing a set of facial expressions. Depending on the initialisation (e.g. face resolution), the system may perform robustly at more than 30 frames per second, in spite of oclussions and illumination changes.

1 Introduction

Automatic face analysis has a broad range of applications ranging from human-computer interaction to video conferencing or computer graphics. A good face analysis system should be non-invasive, that is, the user should not have any artificial marks on his face, nor wear any special device. Given the performance of present desktop computers, the inexpensive cost of digital cameras and the maturity of many of it's techniques, Computer Vision is possibly the most promising approach for buiding an automatic face analysis system.

Although very attractive from an application point of view, a face it is not an easy object to analyse using Computer Vision techiques. This is mainly due to the non-rigid nature of motion of some of its most expressive parts, namely, mouth, jaw, eyes and eyebrows. Tracking algorithms try to represent non-rigid motion by modeling target appearance in various ways. Some use texture [1], colour [2] or shape [3] statistics, or both [4], others employ textured 3D models [5], and finally, many use linear subspace models of texture [6, 7] or shape and texture [8].

In this paper we present a real-time human face expression analysis system for video sequences consisting of head-and-shoulder images. It is based on a tracking architecture made of a set of different tracking algorithms and a control automata which coordinates their execution. This allows the adaptation of the tracker to variations in the environment conditions (illumination, oclussions, etc) by choosing the most appropriate visual cue for tracking. This means that the precision on the estimation of target pose will vary depending on scene conditions.

2 System overview

The system is organised in four levels, or trackers, with a finite automaton co-ordinating their execution (see Fig. 1). Each level represents a tracker with different tracking cues.

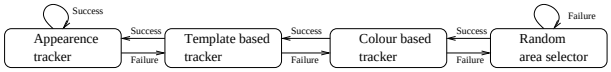


Figure 1: Tracking system state diagram

The first level is a random area selector. When the system is in the initial state or when it has lost the target, it randomly looks for the face in the camera field of view. The second level is a colour-based tracker which uses a modified version of the well known Grey World colour constancy algorithm for modeling skin colour. Next level is a parameterised optical flow-based tracker that estimates the head 3D pose from an initial image template of the user. And finally, the user's facial expressions is estimated through an appearance based tracker that estimates both, pose and the parameters of a linear subspace model of the most expressive parts of the face (the eyes and the mouth).

In the following sections we will present this facial expression analyser.

3 Colour based tracking

Skin colour is the most frequently used feature for face detection and tracking [9]. The primary problem in automatic skin detection is colour constancy. The colour of an image pixel depends not only on the imaged object colour, but also

on the lighting geometry, illuminant colour and camera response. This means that the RGB (Red, Green and Blue) values of a patch of skin can be very different depending on the camera used to capture the image, the colour and intensity of the illumination, the relative orientation between camera, skin surface and light source, or the existence of shadows or highlights in the image. Colour constancy algorithms model these effects and try to obtain colour invariants that facilitate the identification of a given colour under varying environmental conditions. For example [10], if the scene light intensity is scaled by a actor s, then each perceived pixel colour becomes $[sR, sG, sB]$. On the other hand, a change in illuminant colour can be modelled as a scaling α, β and γ in the R, G and B image colour channels. In this case the previous normalisation fails. The Grey World (GW) algorithm [10] provides a constancy solution independent of the illuminant colour by dividing each colour channel by its average value:

$$[\alpha R, \beta G, \gamma B] \mapsto [\frac{\alpha R}{\frac{\alpha}{n}\sum_i R}, \frac{\alpha G}{\frac{\beta}{n}\sum_i G}, \frac{\alpha B}{\frac{\gamma}{n}\sum_i B}].$$

In this section we introduce a colour constancy algorithm that can be used for real-time colour-based image segmentation. The algorithm is based on GW and exploits the redundancy of the image sequence in order to compute the relative change in illumination between the images of the sequence.

3.1 Dynamic GW based tracking

According to the effect on the $[RGB]$ values of geometric and colour changes in the illumination, two pixels $I(ij)$ and $I(kl)$ of an image would have the following $[RGB]$ values: $[s_{ij}\alpha r_{ij}, s_{ij}\beta g_{ij}, s_{ij}\gamma b_{ij}]$, $[s_{kl}\alpha r_{kl}, s_{kl}\beta g_{kl}, s_{kl}\gamma b_{kl}]$, where $[r_{ij}, g_{ij}, b_{ij}]$ and $[r_{kl}, g_{kl}, b_{kl}]$ represent surface reflectance; i.e. real object colour, independent of the illuminant.

Let us define the *image average geometrical reflectance*, $\bar{\mu}$, as

$$\begin{aligned}\bar{\mu} &= [\mu_r, \mu_g, \mu_b] \\ &= \left[\frac{1}{n}\sum_{ij\in I} s_{ij}r_{ij}, \frac{1}{n}\sum_{ij\in I} s_{ij}g_{ij}, \frac{1}{n}\sum_{ij\in I} s_{ij}b_{ij}\right],\end{aligned}$$

where n is the number of image pixels. It represents the average $[RGB]$ image values, once the colour illuminant component has been removed.

If we assume that the average geometrical reflectance is constant over the image sequence, then the following normalisation, called *Projective GW Normalisation*, removes the illuminant colour and intensity changes:

$$\begin{aligned}[s_{ij}\alpha r_{ij}, s_{ij}\beta g_{ij}, s_{ij}\gamma b_{ij}] &\mapsto \left[\frac{\alpha s_{ij} r_{ij}}{\alpha\mu_r}, \frac{\beta s_{ij} g_{ij}}{\beta\mu_g}, \frac{\gamma s_{ij} b_{ij}}{\gamma\mu_b}\right] \\ &\mapsto \left[\frac{s_{ij}r_{ij}}{s_{ij}(\frac{r_{ij}}{\mu_r}+\frac{g_{ij}}{\mu_g}+\frac{b_{ij}}{\mu_b})}, \frac{s_{ij}g_{ij}}{s_{ij}(\frac{r_{ij}}{\mu_r}+\frac{g_{ij}}{\mu_g}+\frac{b_{ij}}{\mu_b})}\right]\end{aligned}$$

This normalisation is still valid just for static scenes. The dynamic extension to the basic GW is based on the fact that when a change in the average geometrical reflectance ($\bar{\mu}$) is detected, the GW colour descriptors for the present image $I^t_{\hat{r}\hat{g}\hat{b}}$ can still be computed with the the average pixel values of the previous image, $\bar{\mu}^{I^{t-1}}_{rgb}$. In this situation we segment the present image with the the average pixel values of the previous one, and let the present image be the reference image.

The problem now is how to detect a change of subsequence. We do this just by searching for a change in the average geometrical reflectance. This cannot be accomplished by analysing $\bar{\mu}^I_{rgb}$, as it also changes with the illuminant colour. We solve this problem by monitoring the average GW descriptors of some face pixels. As they are invariant to illuminant colour changes, a change in these descriptors is necessarily caused by a change in average geometrical reflectance.

4 Template based 3D planar tracking

Let P be the image of a planar object. Assuming no changes in the scene illumination, the following minimisation[1]

$$\min_{\bar{\mu}} ||\mathbf{I}(f(\bar{x}, \bar{\mu}), t_n) - \mathbf{I}(\bar{x}, t_0)||^2, \qquad (1)$$

can be solved linearly by computing $\bar{\mu}$ incrementally while tracking. We can achieve this by making a Taylor series expansion of (1) about $(\bar{\mu}, t_n)$ and computing the increment, $\delta\bar{\mu}$, between two time instants [11]:

$$\delta\bar{\mu} = -(\mathbf{M}^\top\mathbf{M})^{-1}\mathbf{M}^\top[\mathbf{I}(\bar{x}, \bar{\mu}_n) - \mathbf{I}(\bar{x}, \bar{\mu}_0)] \qquad (2)$$

where $\mathbf{M}$ is the Jacobian matrix of the image w.r.t. $\bar{\mu}$ and the dependence of $\mathbf{I}$ on t has been dropped for convenience.

While tracking, matrix $\mathbf{M}$ must be recalculated in each frame, as it depends on $\bar{\mu}$. This is computationally expensive, as $\mathbf{M}$ is of dimension $N \times n$, being N the number of template pixels and n the number of motion parameters. In the sequel we will factor $\mathbf{M}$ in order to simplify this computation [7].

$\mathbf{M}$ can be written as

$$\mathbf{M}(\bar{\mu}) = \begin{bmatrix} \nabla_x I(\bar{x}_1, \bar{\mu}_0)^\top f_x(\bar{x}_1, \bar{\mu})^{-1} f_\mu(\bar{x}_1, \bar{\mu}) \\ \vdots \\ \nabla_x I(\bar{x}_N, \bar{\mu}_0)^\top f_x(\bar{x}_N, \bar{\mu})^{-1} f_\mu(\bar{x}_N, \bar{\mu}) \end{bmatrix}, \qquad (3)$$

[1] $\mathbf{I}(\bar{x}, t)$ is a column vector constructed scanning P.

where $\nabla_x I$ is the template image gradient, f_x is the derivative of the motion model with respect to the pixel coordinates and f_μ is the derivative of the motion model with respect to the motion parameters.

Depending on the motion model, $\mathbf{M}$ may be factored into the product of two matrices [7],

$$\mathbf{M}(\bar{\mu}) = \begin{bmatrix} \nabla_x I(\bar{x}_1, \bar{\mu}_0)^\top \mathbf{\Gamma}(\bar{x}_1) \\ \vdots \\ \nabla_x I(\bar{x}_N, \bar{\mu}_0)^\top \mathbf{\Gamma}(\bar{x}_N) \end{bmatrix} \mathbf{\Sigma}(\bar{\mu}) = \mathbf{M}_0 \mathbf{\Sigma}(\bar{\mu}), \quad (4)$$

a constant matrix $\mathbf{M}_0$ of dimension $N \times m$ and a matrix $\mathbf{\Sigma}$ of dimension $m \times n$, that depends on $\bar{\mu}$. As $\mathbf{M}_0$ can be precomputed, this factorisation reduces the on line computation to

$$\delta\bar{\mu} = -(\mathbf{\Sigma}^\top \mathbf{M}_0^\top \mathbf{M}_0 \mathbf{\Sigma})^{-1} \mathbf{\Sigma}^\top \mathbf{M}_0^\top [\mathbf{I}(\bar{x}, \bar{\mu}_n) - \mathbf{I}(\bar{x}, \bar{\mu}_0)]. \quad (5)$$

Matrix $\mathbf{M}_0$ is the Jacobian of the template image. It is our *a priori* knowledge about target structure, that is, how the grey level value of each pixel changes as the object moves. It represents the information provided by each template pixel to the tracking process. Note that we cannot track any object, as in order to solve (4), a non singular $\mathbf{M}_0^\top \mathbf{M}_0$ matrix is needed. This is a generalisation of the well known *aperture problem* in optical flow computation.

4.1 Projective model for planar tracking

Hager and Belhumeur [7] did the Jacobian matrix factorisation for the translation-scale, affi ne and one non linear motion model. Although is said elsewhere that it is not possible [12], in this section we introduce the factorisation for the projective case.

Let $\bar{x} = (u, v)^\top$ and $\bar{x}_h = (r, s, \lambda)^\top$ be respectively the Cartesian and Projective coordinates of an image pixel. They are related by: $\bar{x}_h = (r, s, \lambda)^\top \rightarrow \bar{x} = (r/\lambda, s/\lambda)^\top = (u, v)^\top$; $\lambda \neq 0$. The 2D projective linear transformation can be written as

$$f(\bar{x}_h, \bar{\mu}) = \mathbf{H}\bar{x}_h = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & 1 \end{bmatrix} \begin{bmatrix} r \\ s \\ \lambda \end{bmatrix},$$

where the motion parameter vector is now $\bar{\mu} = (a, b, c, d, e, f, g, h)^\top$. Taking derivatives of f with respect to $\bar{x}$ and $\bar{\mu}$ [2],

$$f_{\bar{x}_h}(\bar{x}_h, \bar{\mu})^{-1} = H^{-1}, \quad (6)$$

$$f_{\bar{\mu}}(\bar{x}_h, \bar{\mu}) = \begin{bmatrix} r & 0 & 0 & s & 0 & 0 & \lambda & 0 \\ 0 & r & 0 & 0 & s & 0 & 0 & \lambda \\ 0 & 0 & r & 0 & 0 & s & 0 & 0 \end{bmatrix} \quad (7)$$

[2] The derivative of $\mathbf{I}$ w.r.t. $\bar{x}_h$ is $\nabla_{\bar{x}_h} \mathbf{I}(\bar{x}_h) = \left[\frac{\partial \mathbf{I}}{\partial u}, \frac{\partial \mathbf{I}}{\partial v}, -u\frac{\partial \mathbf{I}}{\partial u} - v\frac{\partial \mathbf{I}}{\partial v}\right]^\top$

and introducing them into equation (4) the projective factorisation of $\mathbf{M}$ arises:

$$\Gamma(\bar{x}_h) = \begin{bmatrix} rI_{3\times3} & | & sI_{3\times3} & | & tI_{3\times3} \end{bmatrix},$$
$$\Sigma(\bar{\mu}) = \begin{bmatrix} H^{-1} & 0 & 0 \\ 0 & H^{-1} & 0 \\ 0 & 0 & H_{12}^{-1} \end{bmatrix}.$$

Note that the planar tracking algorithm has to iterate over the equation (5) until convergence. In the current implementation we make the processing at two resolution levels and two iterations per level at video frame rate.

4.2 3D pose estimation

The tracking model presented in the previous section computes the homography $\mathbf{H}_0^n$ between the present image and the stored template. In this section we are going to show that it is possible to estimate the 3D pose of the tracked patch from $\mathbf{H}_0^n$ by using a calibrated vision system.

So far we have only computed 2D information. In order to have 3D information we have to compute two more homographies: one from P to the image plane at $\bar{t}_0$, $\mathbf{H}_W^0$, and another from P to the image plane at $\bar{t}_n$, $\mathbf{H}_W^n$, see Fig. 2.

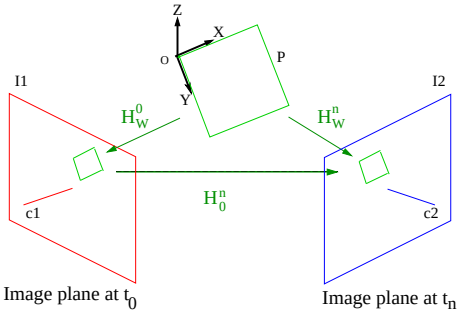


Figure 2: Projective transformations involved in 3D plane tracking.

Let $(X_P, Y_P)^\top$ be the Cartesian coordinates of a known point in P and let $(x_0, y_0)^\top$ be Cartesian coordinates of the projection of $(X_P, Y_P)^\top$ onto I at t_0 (i.e. at the template image). $\mathbf{H}_W^0$ can be computed off line from:

$$[x_0 \; y_0 \; 1]^\top = \mathbf{H}_W^0 \, [X_P \; Y_P \; 1]^\top. \quad (8)$$

given the projection of, at least, four known points on P.

On the other hand, $\mathbf{H}_W^n$ can be computed from the camera projection matrix by choosing the scene coordinate system to have X and Y axes on plane P and axis Z perpendicular to it (see Fig. 2).

$$[x_n \; y_n \; 1]^\top = \underbrace{\lambda \mathbf{K}[\bar{r}_1 \; \bar{r}_2 \; \bar{t}]}_{\mathbf{H}_W^n} [X_P \; Y_P \; 1]^\top \quad (9)$$

where $\mathbf{R}$ and $\bar{t}$ are respectively the orientation and the position of P in the camera coordinate system, $\bar{r}_i$ is the ith column of matrix $\mathbf{R}$, λ is a scale factor and $\mathbf{K}$ is the camera intrinsics matrix.

Now, from (8) and (9) we get

$$[x_n\ y_n\ 1]^\top = \underbrace{\lambda \mathbf{K}[\bar{r}_1\ \bar{r}_2\ \bar{t}](\mathbf{H}_W^0)^{-1}}_{\mathbf{H}_0^n} [x_0\ y_0\ 1]^\top . \quad (10)$$

From which we obtain the relation between the homography computed in the previous section, $\mathbf{H}_0^n$, and the pose of P. So, if the intrinsics $\mathbf{K}$ and the homographies $\mathbf{H}_W^0$ and $\mathbf{H}_0^n$ are known, we can compute $\mathbf{H}^*$,

$$\mathbf{H}^* = \mathbf{K}^{-1}\mathbf{H}_0^n\mathbf{H}_W^0 = \lambda[\bar{r}_1\ \bar{r}_2\ \bar{t}] \quad (11)$$

The translation is obtained directly from the third column of $\mathbf{H}^*$ but in order to obtain the rotation matrix we still have to impose some constraints:

- $\| \bar{r}_1 \| = \| \bar{r}_2 \| = 1$, as $\mathbf{R}$ is a rotation matrix. In this way we get $\widehat{r}_1$ and $\widehat{r}_2$.
- $\bar{r}_3 \perp \bar{r}_1$ and $\bar{r}_3 \perp \bar{r}_2$, from where we get $\widehat{r}_3$.

4.3 Pixel selection for faster tracking

In this section we will further improve the tracking procedure presented in the previous section by reducing the number of template pixels used for solving equation (5). This improvement comes not only from having a smaller matrix $\mathbf{M}_0$, but mainly from diminishing the number of pixels warped to compute $I(\bar{x}, \bar{\mu}_n)$.

The Jacobian matrix $\mathbf{M}$ of image I can be expressed as:

$$\mathbf{M} = (I_{\bar{\mu}_1}, I_{\bar{\mu}_2}, \cdots, I_{\bar{\mu}_n}), \quad (12)$$

where $I_{\bar{\mu}_i} = \frac{\partial I(\bar{x},\bar{\mu})}{\partial \bar{\mu}_i}$ is a column vector with an entry for every pixel in I. It represents the changes in image brightness induced by motion $\bar{\mu}_i$ (see Fig. 3). Thus, $\mathbf{M}$ relates variations in motion parameters to variations in brightness values. Note that equation (5) works in the opposite direction, i.e. it uses $\mathbf{M}$ to compute motion from observed changes in brightness values. Let us call $I_{\bar{\mu}}^\top(\bar{x})$ the row in

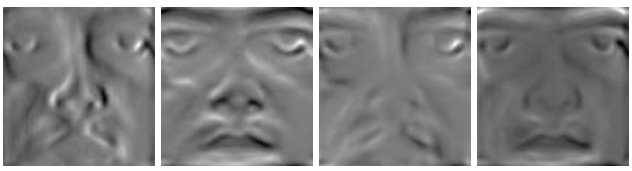


Figure 3: Jacobian matrix for a translation (x, y), rotation (θ) and scale (s) motion model. In reading direction each image represents respectively I_x, I_y, I_θ, I_s.

$\mathbf{M}$ corresponding to image pixel $I(\bar{x})$. Each row entry is the derivative of image pixel $I(\bar{x})$ with respect to a model parameter $\bar{\mu}_i$ ($\forall i = 1 \ldots n$). Intuitively, a pixel with a small $||I_{\bar{\mu}}(\bar{x})||$ provides almost no information for solving (5). So, a good pixel for tracking is one with a large $||I_{\bar{\mu}}(\bar{x})||$. Given two image pixels $I(\bar{x}_1)$ and $I(\bar{x}_2)$, one of them is redundant if $I_{\bar{\mu}}(\bar{x}_1) \approx I_{\bar{\mu}}(\bar{x}_2)$. So, a good set of pixels for tracking is one such that $\mathbf{M}^\top\mathbf{M}$ is not singular.

Selecting the "best" set of p pixels is a combinatorial search problem, as all $\binom{p}{N}$ sets of pixels should be considered in order to select the most informative one. In the context of image registration, Dellaert selects p pixels randomly from the top 20% of pixels with highest $||I_{\bar{\mu}}(\bar{x})||$ [13]. In our experiments we have found that the best set of pixels for tracking is the one with highest $||I_{\bar{\mu}}(\bar{x})||$, lowest redundancy and most even distribution on the image. In the sequel we will present a procedure to select a set of pixels with high $||I_{\bar{\mu}}(\bar{x})||$ and low redundancy.

If we consider each row vector $I_{\bar{\mu}}(\bar{x})$ as a point in n-dimensional space, then the points in the convex hull of this cloud are those with highest $||I_{\bar{\mu}}(\bar{x})||$ and lowest redundancy. Let us call this set of points the *Jacobian cloud*. Computing the convex hull of a Jacobian cloud with thousands of points in a 8-dimensional space (the projective motion model has 8 parameters) can be time consuming. On the other hand, as can be seen in Fig. 4 (right), the distribution of points for this model is highly correlated, with two space directions representing 99.96% of the total variance in the cloud. So, a good approximation to the convex hull of the cloud would be to compute the convex hull of its projection onto the two main directions (see Fig. 4, left). If we

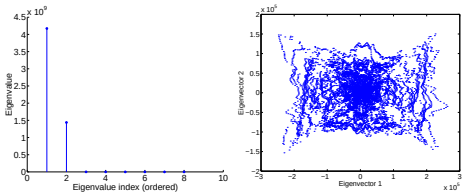


Figure 4: Eigenvalues of the Jacobian cloud's covariance matrix (left) and view of the projection of the Jacobian cloud onto the two principal directions (right).

choose the points from the outer convex hulls (like peeling off an orange) then only the pixels in the strongest edges of the image would be selected. In order to achieve a more even spatial distribution of the selected pixels we choose all pixels of a randomly selected set of convex hulls from the outer 30% of them (see Fig. 5).

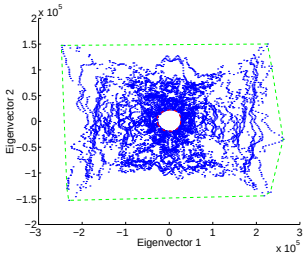


Figure 5: Points in the outer 30% convex hulls in projected space.

5 Appearance based tracking

Let P be the image of a target. Appearance based tracking consists on estimating for each image in the sequence the values of the motion, $\bar{\mu}$, and appearance, $\bar{c}$, parameters which minimise the error function

$$E(\bar{\mu}, \bar{c}) = ||I(f(\bar{x}, \bar{\mu}), t) - [\mathbf{B}\bar{c}(t)](\bar{x})||^2, \qquad (13)$$

where $\bar{x}$ is the vector of co-ordinates of a point in image I, $\mathbf{B}$ is the PCA subspace base matrix, $\bar{c}$ is the vector of PCA coefficients, and $I(f(\bar{x}, \bar{\mu}), t)$ is the image acquired at time t rectified with motion model $f(\bar{x}, \bar{\mu})$ and motion parameters $\bar{\mu}$. By $[\mathbf{B}\bar{c}](x)$ we denote the value of $\mathbf{B}\bar{c}$ for the pixel with position $\bar{x}$ in the image. Matrix $\mathbf{B}$ is of dimension $N \times k$, where N is the number of pixels per image and k is the number of basis vectors.

In general, minimising (13) can be a difficult task as it defines a non-convex objective function. Several procedures have been proposed to solve this problem which can be grouped into those using gradient descent [6] and those using Gauss-Newton iterations [12, 7, 14]. Black and Jepson [6] presented an iterative solution by using a gradient descent procedure and a robust metric with increasing resolution levels. Their algorithm lacked efficiency as, for example, the Jacobian of each incoming image had to be computed once on every frame for each level in the multiresolution pyramid.

We have developed a real-time procedure for minimising equation (13) [15]. It is based on a factorisation of $\mathbf{M}$, similar to 4. By using it, we can make a modular eigentracking [16] of the eyes and the mouth at 30 frames per second.

6 Experiments

The current implementation of the system has been tested extensively in our lab in a great variety of conditions. In this section we are going to show how the system performs in a difficult sequence in which the user moves and performs facial expressions at the same time (See Fig. 6).

The visual results of the tracking process are shown in Fig. 6. Overlayed over each image are the area of interest of each of the algorithms used. In yellow (and only in the first image) is the random search result, in white, with a cross in the centre,is the result of colour-based tracking and, the difference from the template based from the appearance based tracker, is one red quadrangle over the face versus three red quadrangles over the eyes and mouth. ⇐ ??

Figure 6: Full system processing results. In this case the system works at 30 frames per second processing 1668 images stored on disk. The system performs using the best algorithm for each situation.

The system has been calibrated in different illumination conditions (and even with a different camera) to the test sequence. It accurately tracks the face from the beginning of the sequence (see first frames in fig. 7 processed in the random search state) until the user is too close to the camera to correctly track him (see last frames in fig. 7). This can be validated from the fact that the lower layer that is used, apart from the beginning and the end of the sequence, is the colour tracker as shown in Fig. 7. ⇐ ?? ⇐ ??

7 Conclusions

In this paper we have presented a facial analysis solution based on the cooperation of a set of trackers. This allows the adaptation of the system to the environment conditions by choosing the best tracker at each time instant. With the current implementation we can track a face at 30 frames per

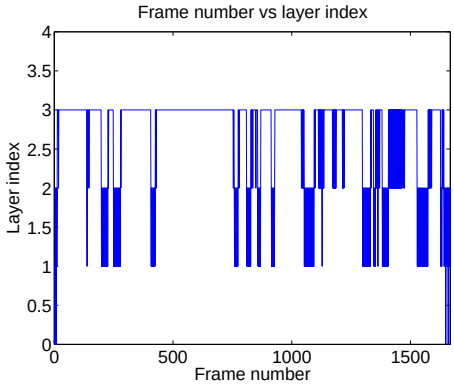


Figure 7: Full system processing layers. The layer index 0 represents the random area search, 1 the colour tracker, 2 the template based tracker and 3 the appearance based tracker.

second achieving video-rate performance.

The system is not yet finished. Now we are working on the appearance to animation parameters translation problem and on introducing efficient illumination invariance within the appearance-based tracking paradigm.

Acknowledgements

Work funded by CICyT under project number TIC1999-1021 and TIC2002-00591.

References

[1] A. Jepson, D. Fleet, and T. El-Maraghi, "Robust online appearance models for visual tracking," in *Proc. of Int. Conf. on Computer Vision and Pattern Recognition*, vol. I. IEEE, 2001, pp. 415–422.

[2] D. Comaniciu, V. Ramesh, and P. Meer, "Real-tiem tracking of non-rigid objects using mean shift," in *Proc. of Int. Conf. on Computer Vision and Pattern Recognition*. IEEE, 2000, pp. 142–149.

[3] M. Isard and A. Blake, "Condensation–conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 2–28, 1998.

[4] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proc, of the Int. Conf. on Computer Vision and Pattern Recognition*. IEEE, 1998, pp. 232–237.

[5] M. La Cascia, S. Sclaroff, and V. V. Athitsos, "Fast, reliable head tracking under varying illumination: An approach based on robust registration of texture-mapped 3d models," *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, no. 4, pp. 322–336, April 2000.

[6] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[7] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Transactions on Pattern Analisys and Machine Intelligence*, vol. 20, no. 10, pp. 1025–1039, 1998.

[8] T. Cootes, G. Edwards, and C. Taylor, "Active appearance models," in *Proc. European Conference on Computer Vision*. Springer-Verlag, 1998.

[9] K. Toyama, "Prolegomena for robust face tracking," Microsoft Research, Tech. Rep. MSR-TR-98-65, Nov. 1998.

[10] B. S. Graham D. Finlayson and J. L. Crowley, "Comprehensive colour image normalization," in *Proc. Fith European Conference on Computer Vision, ECCV'98*, vol. I. IEEE, 1998, pp. 475–490.

[11] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. of Imaging Understanding Workshop*, 1981, pp. 121–130.

[12] S. Baker and I. Matthews, "Equivalence and efficiency of image alignment algorithms," in *Proc. of International Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2001, pp. I–1090–I–1097.

[13] F. Dellaert and R. Collins, "Fast image-based tracking by selective pixel integration," in *ICCV99 Workshop on frame-rate applications*. IEEE, 1999.

[14] F. Jurie and M. Dhome, "Hyperplane approximation for template matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 996–100, 2002.

[15] J. Buenaposada, E. Muñoz, and L. Baumela, "Efficient appearance-based tracking," in *Proc. of CVPR'2004*, vol. Submitted, 2004.

[16] F. de la Torre and M. Black, "Robust parameterized component analysis," in *Proc. European Conference on Computer Vision, LNCS 2353*. Springer-Verlag, 2002, pp. 653–669.

Some experiments on incremental image alignment

Enrique Muñoz, José Miguel Buenaposada, Luis Baumela
Departamento de Inteligencia Artificial
Facultad de Informática, Univ. Politécnica de Madrid
Campus de Montegancedo s/n
28660 Boadilla del Monte, Madrid, Spain
Email: {kike,jmbuena}@dia.fi.upm.es, lbaumela@fi.upm.es

Abstract

Incremental image alignment is a research topic which has recently received very much atention for its applications in real-time and deformable object tracking. In the paper we compare some additive and compositional incremental alignment procedures pointing out their limitations and strengths. We also present and compare an alternative approximate additive procedure which should be further refined in order to be useful for tracking.

1 Introduction

Tracking planar patches is a subject of interest in computer vision, with applications in augmented reality [**?**], mobile robot navigation [**?**], face tracking [**?**], or the generation of super-resolution images [**?**], to name a few. One of the most usual approaches for patch tracking is based on incremental registration. This is achieved by defining an image warping function and by minimising image discrepancies through a non linear cost function. Different approaches to solve the minimisation problem have been proposed in the literature depending on the minimisation procedure, on whether the variation in model parameters are additively or compositionally updated or on which term of the cost function the linear approximation is made. In this paper we will compare some of these approaches.

In the following section we will briefly introduce the incremental registration problem and the associated cost function. In sections 3 to 8 we present different approaches for minimising the mentioned cost function. Finally, in section 9 we make some experimental comparisons and in section 10 we draw conclusions.

2 Incremental image alignment

Let $\mathbf{x}$ represent the location of a point in an image and $I(\mathbf{x}, t)$ represent the brightness value of that location in the image acquired at time t. Let $\mathcal{R} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ be a set of N image points of the object to be tracked (*target region*), whose brightness values are known in a reference image $I_r(\mathbf{x})$. These image points together with their brightness values at the reference image represent the *reference template* to be tracked.

As the target moves, the relative motion between camera and object causes the target region to shift and deform in the image. Let us represent this deformation by a *motion model* $\mathbf{f}(\mathbf{x}, \boldsymbol{\mu})$ parametrised by $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_n)^\top$, with $N > n$ and $\mathbf{f}$ differentiable both in $\mathbf{x}$ and $\boldsymbol{\mu}$. Let the vector $\boldsymbol{\mu}_I$ represent the set of parameters of the identity transformation, $\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_I) = \mathbf{x}$, and the vector $\boldsymbol{\mu}_0$ represent the position and deformation of the target region in the initial image of the sequence, $I_r(\mathbf{x}) = I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_0), t_0)$, which we will assume to be known.

Finally, let us also assume that variations in brightness values are only caused by image motion, i.e. that the *image constancy assumption* holds for all pixels in $\mathcal{R}$

$$I_r(\mathbf{x}) = I_r(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_I)) = I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t^*), t) \;\; \forall \mathbf{x} \in \mathcal{R}, \qquad (1)$$

where $\boldsymbol{\mu}_t^*$ is the actual image deformation at time t.

For us, tracking the object means recovering the motion parameter vector of the target region for each image in the sequence. This can be achieved by minimising the following least-squares objective function for every image in the sequence

$$\min_{\boldsymbol{\mu}} \parallel \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t) - \mathbf{I}_r(\mathbf{x}) \parallel^2 \qquad (2)$$

where $\mathbf{I}(\mathbf{x})$ is a column vector with all the pixel values in image $I(\mathbf{x})$.

In general, (2) is a difficult minimisation problem, which cannot be linearly solved. In the computer vision literature it has been traditionally computed by using the continuity of motion to estimate a starting point for the minimisation [**?**, **?**, **?**, **?**]. If at some initial time, t_0, the parameters of the motion model, $\boldsymbol{\mu}_0$, are known, then (2) can be locally linearised by making a Taylor series expansion at $(\boldsymbol{\mu}_0, t_0)$.

In this way, the position of the target region at time $t_0 + \delta t$ can be linearly estimated, provided the displacement of the target region between both time instants, represented by a variation $\delta\boldsymbol{\mu}$ in the model parameters, is small. By repeating this process for every new image in the sequence, the model parameters $\boldsymbol{\mu}_t$ at any time $t > t_0$ can be estimated.

3 Lucas and Kanade's Algorithm

Here we describe the alignment procedure introduced by Lucas and Kanade in their seminal work [**?**]. Let us assume that we have an estimate of the motion model parameters at time t, i.e. we have some $\boldsymbol{\mu}_t$ such that $I_r(\mathbf{x}) \approx I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t)$. The tracking problem can be stated as that of estimating from $I(\mathbf{z}, t + \delta t)$, an offset $\delta\boldsymbol{\mu}_a$ in the motion parameters such that

$$I_r(\mathbf{x}) \approx I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_{t+\delta t}), t+\delta t) = I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t + \delta\boldsymbol{\mu}_a), t+\delta t).$$

Note that the increment in the model parameters is additively composed with the previous parameter estimates, i.e. $\boldsymbol{\mu}_{t+\delta t} = \boldsymbol{\mu}_t + \delta\boldsymbol{\mu}_a$.

Then the minimisation in (2) can be rewritten as

$$\min_{\delta\boldsymbol{\mu}_a} \parallel \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t + \delta\boldsymbol{\mu}_a), t + \delta t) - \mathbf{I}_r(\mathbf{x}) \parallel^2 . \quad (3)$$

This equation can be linearised by making a Taylor series expansion at the $(\boldsymbol{\mu}_t, t)$ The following minimisation results

$$\min_{\delta\boldsymbol{\mu}_a} \parallel \mathbf{M}(\boldsymbol{\mu}_t, t)\delta\boldsymbol{\mu}_a + \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t) - \mathbf{I}_r(\mathbf{x}) \parallel^2, \quad (4)$$

where $\mathbf{M}(\boldsymbol{\mu}_t, t)$ is the Jacobian of the image values in the target region w.r.t. the motion parameters:

$$\mathbf{M}(\boldsymbol{\mu}_t, t) = \begin{pmatrix} \frac{\partial \mathbf{I}(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}), t)}{\partial \boldsymbol{\mu}}\Big|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \\ \vdots \\ \frac{\partial \mathbf{I}(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}), t)}{\partial \boldsymbol{\mu}}\Big|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \end{pmatrix}. \quad (5)$$

This equation can be solved posing it as follows (note that it is linear in the unknowns $\delta\boldsymbol{\mu}_a$):

$$\mathbf{M}(\boldsymbol{\mu}_t, t)\delta\boldsymbol{\mu}_a \; = \; \boldsymbol{\varepsilon}(t + \delta t), \quad (6)$$

where $\boldsymbol{\varepsilon}(t + \delta t) = \mathbf{I}_r(\mathbf{x}) - \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t)$. Parameters increment $\delta\boldsymbol{\mu}_a$ can be computed in a least-squares sense:

$$\delta\boldsymbol{\mu}_a = (\mathbf{M}(\boldsymbol{\mu}_t, t)^\top \mathbf{M}(\boldsymbol{\mu}_t, t))^{-1} \mathbf{M}(\boldsymbol{\mu}_t, t)\boldsymbol{\varepsilon}(t + \delta t), \quad (7)$$

$\mathbf{M}(\boldsymbol{\mu}_t, t)$ can be estimated from the image gradients at instant t:

$$\mathbf{M}(\boldsymbol{\mu}_t, t) = \begin{pmatrix} \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}_t), t)^\top \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_1, \boldsymbol{\mu}_t) \\ \vdots \\ \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}_t), t)^\top \mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}_N, \boldsymbol{\mu}_t) \end{pmatrix}, \quad (8)$$

$$\mathbf{f}_{\boldsymbol{\mu}}(\mathbf{x}, \boldsymbol{\mu}) = \left(\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}_1}\Big|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} | \ldots | \frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}_n}\Big|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \right), \quad (9)$$

The main limitation of this algorithm, when used for tracking, is the computational cost of computing Jacobian vector for each pixel in $\mathcal{R}$ with respect to $\boldsymbol{\mu}$.

4 Hager and Belhumeur's Algorithm

Hager and Belhumeur proposed an efficient procedure to compute the image Jacobians [**?**]. First they avoid the computation of $\nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t)$ for each image in the sequence. This is achieved by expressing it in terms of the gradient of the reference template. If we assume that the present estimation of the motion parameters is exact, $\boldsymbol{\mu}_t = \boldsymbol{\mu}_t^*$, then differentiating (1) gives us:

$$\nabla_{\mathbf{x}} I_r(\mathbf{x})^\top \left(\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t)}{\partial \mathbf{x}} \right)^{-1} = \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t)^\top. \quad (10)$$

The partial derivatives of the image values w.r.t. the motion parameters can be now expressed as:

$$\frac{\partial I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}), t)}{\partial \boldsymbol{\mu}} = \nabla_{\mathbf{x}} I_r(\mathbf{x})^\top \left(\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu})}{\partial \mathbf{x}} \right)^{-1} \frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}}, \quad (11)$$

which means that $\mathbf{M}(\boldsymbol{\mu}, t)$, at any time instant, depends only on the motion model and on the spacial gradients of the reference template (which are constant during all the sequence). So $\mathbf{M}(\boldsymbol{\mu}, t)$ will be independent of t and we will denote it $\mathbf{M}(\boldsymbol{\mu})$. Unfortunately, in general, the other two terms in equation (11), $\mathbf{f}_{\mathbf{x}}^{-1}$ and $\mathbf{f}_{\boldsymbol{\mu}_t}$, are not constant. However, if $\mathbf{f}$ is chosen such that

$$\left(\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t)}{\partial \mathbf{x}} \right)^{-1} \left(\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}} \right)_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} = \boldsymbol{\Gamma}(\mathbf{x})\boldsymbol{\Sigma}(\boldsymbol{\mu}), \quad (12)$$

where $\boldsymbol{\Gamma}(\mathbf{x})$ is a matrix that depends only on the position of the pixel and $\boldsymbol{\Sigma}(\boldsymbol{\mu})$ is a matrix that depends only on the motion parameters, then we can write $\mathbf{M}$ as

$$\mathbf{M}(\boldsymbol{\mu}) = \begin{pmatrix} \nabla_{\mathbf{f}} I_r(\mathbf{x}_1)^\top \boldsymbol{\Gamma}(\mathbf{x}) \\ \vdots \\ \nabla_{\mathbf{f}} I_r(\mathbf{x}_N)^\top \boldsymbol{\Gamma}(\mathbf{x}) \end{pmatrix} \boldsymbol{\Sigma}(\boldsymbol{\mu}) = \mathbf{M}_0 \boldsymbol{\Sigma}(\boldsymbol{\mu}) \quad (13)$$

where $\mathbf{M}_0$ is constant.

Now the least-squares solution is of the form

$$\delta\boldsymbol{\mu}_a = (\boldsymbol{\Sigma}(\boldsymbol{\mu})^\top \boldsymbol{\Lambda} \boldsymbol{\Sigma}(\boldsymbol{\mu})]^{-1} \boldsymbol{\Sigma}(\boldsymbol{\mu})^\top \mathbf{M}_0^\top \boldsymbol{\varepsilon}(\mathbf{x}, t + \delta t). \quad (14)$$

where $\boldsymbol{\varepsilon}(t + \delta t) = \mathbf{I}_r(\mathbf{x}) - \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t)$ and $\boldsymbol{\Lambda} = \mathbf{M}_0 \top \mathbf{M}_0$ is a constant matrix, which can be precomputed offline. The size of this matrix depends on the factorisation (12), but in general it is small and related to n. In this

case the on-line computation performed by this algorithm is much smaller than the original one proposed by Lucas and Kanade. The main limitation comes from the factorisation of (12) which, in general, is not possible.

5 Szelisky's compositional algorithm

Tracking problem can be introduced in a different way. R. Szeliski proposed in [**?**] a compositional approach in which we can find an update $\delta\boldsymbol{\mu}_c$ to the motion parameters in the following way:

$$I_r(\mathbf{x}) \approx I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_{t+\delta t}), t+\delta t) = I(\mathbf{f}(\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}_I + \delta\boldsymbol{\mu}_c), \boldsymbol{\mu}_t), t+\delta t), \tag{15}$$

where $\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}_I) = \mathbf{x}$. Note that we write the composition with a different function $\mathbf{g}$ to highlight the fact that it can be different from $\mathbf{f}$.

Then the minimisation in (2) can be rewritten as

$$\min_{\delta\boldsymbol{\mu}_c} \parallel \mathbf{I}(\mathbf{f}(\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}_I + \delta\boldsymbol{\mu}_c), \boldsymbol{\mu}_t), t+\delta t) - \mathbf{I_r}(\mathbf{x}) \parallel^2, \tag{16}$$

Using a first order Taylor series expansion at $(\boldsymbol{\mu}_I, t)$ we get

$$\min_{\delta\boldsymbol{\mu}_c} \parallel \mathbf{M}_{cs}(\boldsymbol{\mu}_t, t)\delta\boldsymbol{\mu}_c + \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t+\delta t) - \mathbf{I}_r(\mathbf{x}) \parallel^2, \tag{17}$$

which is linear in the unknowns $\delta\boldsymbol{\mu}_c$ and can be solved by least-squares

$$\delta\boldsymbol{\mu}_c = (\mathbf{M}_{cs}(\boldsymbol{\mu}_t, t)^\top \mathbf{M}_{cs}(\boldsymbol{\mu}_t, t))^{-1} \mathbf{M}_{cs}(\boldsymbol{\mu}_t, t)\boldsymbol{\varepsilon}(t+\delta t), \tag{18}$$

where $\boldsymbol{\varepsilon}(t+\delta t) = \mathbf{I}_r(\mathbf{x}) - \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t+\delta t)$. $\mathbf{M}_{cs}(\boldsymbol{\mu}_t, t)$ can be computed from the gradients of the image values in the target region

$$\mathbf{M}_{cs}(\boldsymbol{\mu}_t, t) = \begin{pmatrix} \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}_t), t)^\top \mathbf{g}_{\boldsymbol{\mu}}(\mathbf{x}_1, \boldsymbol{\mu}_t) \\ \vdots \\ \nabla_{\mathbf{f}} I(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}_t), t)^\top \mathbf{g}_{\boldsymbol{\mu}}(\mathbf{x}_N, \boldsymbol{\mu}_t) \end{pmatrix}, \tag{19}$$

$$\mathbf{g}_{\boldsymbol{\mu}}(\mathbf{x}, \boldsymbol{\mu}) = \left(\left.\frac{\partial \mathbf{g}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}_1}\right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} | \dots | \left.\frac{\partial \mathbf{g}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}_n}\right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \right). \tag{20}$$

Comparing the Shum and Szeliski algorithm [**?**] with the Lucas and Kanade procedure [**?**], we avoid the computation of the Jacobian of the motion model (that can be precomputed) but we still need to compute the gradient of $I(\mathbf{y}, t)$ for each image in the sequence. So, this algorithm is not as efficient as the Hager and Belhumeur approach [**?**]. Another limitation is that the set of motion models that can be used must verify $\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_{t+\delta t}) = \mathbf{f}(\mathbf{g}(\mathbf{x}, \boldsymbol{\mu}_I + \delta\boldsymbol{\mu}_c), \boldsymbol{\mu}_t)$.

6 Baker's forward compositional algorithm

In [**?**], S. Baker proposed a compositional approach in which the $\mathbf{g}$ function is equal to $\mathbf{f}$. The minimisation in (2) can be rewritten as

$$\min_{\delta\boldsymbol{\mu}_c} \parallel \mathbf{I}(\mathbf{f}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_I + \delta\boldsymbol{\mu}_c), \boldsymbol{\mu}_t), t+\delta t) - \mathbf{I}_r(\mathbf{x}) \parallel^2, \tag{21}$$

where $\boldsymbol{\mu}_I$ is the identity for function $\mathbf{f}$. Performing a first order Taylor series expansion at $(\boldsymbol{\mu}_I, t)$ we get

$$\min_{\delta\boldsymbol{\mu}_c} \parallel \mathbf{M}_c(\boldsymbol{\mu}_I, t)\delta\boldsymbol{\mu}_c + \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t+\delta t) - \mathbf{I}_r(\mathbf{x}) \parallel^2, \tag{22}$$

which is linear in the unknowns $\delta\boldsymbol{\mu}_c$. It can be solved using least-squares

$$\boldsymbol{\mu}_c = (\mathbf{M}_{cb}(\boldsymbol{\mu}_t, t)\mathbf{M}_{cb}(\boldsymbol{\mu}_t, t)^\top)^{-1} \mathbf{M}_{cb}(\boldsymbol{\mu}_t, t)\boldsymbol{\varepsilon}(t+\delta t), \tag{23}$$

where $\boldsymbol{\varepsilon}(t+\delta t) = \mathbf{I}_r(\mathbf{x}) - \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t+\delta t)$.

As with the Shum and Szeliski's algorithm we avoid the computation of the Jacobian of the image (which can be precomputed) but we still need to compute the gradient of $I(\mathbf{y}, t)$ for each image in the sequence.

7 Baker's Inverse compositional algorithm

There is a second compositional algorithm proposed by Baker in [**?**]. This is, in contrast with the forward compositional , as efficient as the Hager and Belhumeur's method . In this case, the term in 2 is expanded.

Then, the minimisation in (2) can be rewritten as

$$\min_{\delta\boldsymbol{\mu}_c} \parallel \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t+\delta t) - \mathbf{I}_r(\mathbf{f}^{-1}(\mathbf{x}, \boldsymbol{\mu}_I + \delta\boldsymbol{\mu}_c)) \parallel^2 . \tag{24}$$

Performing a first order Taylor series expansion of the reference template at $(\boldsymbol{\mu}_I, t)$ we obtain

$$\mathbf{I}_r(\mathbf{f}^{-1}(\mathbf{x}, \boldsymbol{\mu}_I + \delta\boldsymbol{\mu}_c)) = \mathbf{I}_r(\mathbf{f}^{-1}(\mathbf{x}, \boldsymbol{\mu}_I)) + \mathbf{M}_c(\boldsymbol{\mu}_I)\delta\boldsymbol{\mu}_c + h.o.t. \tag{25}$$

The variance of the template brightness values w.r.t. the motion parameters is represented by:

$$\mathbf{M}_c(\boldsymbol{\mu}_I) = \begin{pmatrix} \frac{\partial I_r(\mathbf{f}^{-1}(\mathbf{x}_1, \boldsymbol{\mu}))}{\partial \boldsymbol{\mu}} \\ \vdots \\ \frac{\partial I_r(\mathbf{f}^{-1}(\mathbf{x}_N, \boldsymbol{\mu}))}{\partial \boldsymbol{\mu}} \end{pmatrix}. \tag{26}$$

Note that this Jacobian only depends on the motion parameters at the initial instant, $\boldsymbol{\mu}_I$. So, $\mathbf{M}_c$ is a constant matrix. Equation (25) can be solved for the unknowns $\delta\boldsymbol{\mu}_c$ in a least-squares sense as follows

$$\delta\boldsymbol{\mu}_c = (\mathbf{M}_c\mathbf{M}_c^\top)^{-1}\mathbf{M}_c\boldsymbol{\varepsilon}(t+\delta t). \tag{27}$$

$\mathbf{M}_c$ can be computed as follows

$$\mathbf{M}_c = \begin{pmatrix} \nabla_{\mathbf{x}} I_r(\mathbf{x}_1)^\top \mathbf{f}_{\boldsymbol{\mu}}^{-\mathbf{1}}(\mathbf{x}_1, \boldsymbol{\mu}) \\ \vdots \\ \nabla_{\mathbf{x}} I_r(\mathbf{x}_N)^\top \mathbf{f}_{\boldsymbol{\mu}}^{-\mathbf{1}}(\mathbf{x}_N, \boldsymbol{\mu}) \end{pmatrix}, \quad (28)$$

$$\mathbf{f}_{\boldsymbol{\mu}}^{-\mathbf{1}}(\mathbf{x}, \boldsymbol{\mu}) = \left(\frac{\partial \mathbf{f}^{-1}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}_1} | \dots | \frac{\partial \mathbf{f}^{-1}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}_n} \right), \quad (29)$$

and $\boldsymbol{\varepsilon}(t + \delta t) = \mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_t), t + \delta t) - \mathbf{I}_r(\mathbf{x})$.

With this algorithm the Jacobian can be precomputed, but we need a motion model, $\mathbf{f}$, being closed under composition, invertible and having a parameter vector for the identity transformation, $\boldsymbol{\mu}_I$. That is, we are looking for motion models that conform a group.

8 Approximate Additive Algorithm

Assuming that the *brightness constancy constraint* holds, we will derive a new assumption by just differentiating both sides of (1) with respect to the motion parameters:

$$\left. \frac{\partial I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}), t_0)}{\partial \boldsymbol{\mu}} \right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_0} \approx \left. \frac{\partial I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}), t)}{\partial \boldsymbol{\mu}} \right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t}, \quad (30)$$

where $\mathbf{I}(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}), t_0) = \mathbf{I}_r(\mathbf{x})$. Now we can then re-write the Jacobian of the image with respect to the motion parameters in terms of the gradient of the template image:

$$\mathbf{M}(\boldsymbol{\mu}, t) = \begin{pmatrix} \left. \frac{\partial \mathbf{I}(\mathbf{f}(\mathbf{x}_1, \boldsymbol{\mu}), t_0)}{\partial \boldsymbol{\mu}} \right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_0} \\ \vdots \\ \left. \frac{\partial \mathbf{I}(\mathbf{f}(\mathbf{x}_N, \boldsymbol{\mu}), t_0)}{\partial \boldsymbol{\mu}} \right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_0} \end{pmatrix}.$$

The partial derivatives of $\mathbf{I}$ with respect to $\boldsymbol{\mu}$ cannot be evaluated directly. Applying the chain rule we have:

$$\left. \frac{\partial I(\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}), t_0)}{\partial \boldsymbol{\mu}} \right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_0} = \left. \frac{\partial \mathbf{I}(\mathbf{u}, t_0)}{\partial \mathbf{u}} \right|_{\mathbf{u}=\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_0)} \left. \frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\mu})}{\partial \boldsymbol{\mu}} \right|_{\boldsymbol{\mu}=\boldsymbol{\mu}_0}.$$

Note that the gradient of the template image is computed with respect to its coordinate frame, and *warped* using $\mathbf{f}(\mathbf{x}, \boldsymbol{\mu}_0)$. Also note that the derivatives of this Jacobian are evaluated at t_0 and for the initial motion parameters. Therefore $\mathbf{M}(\boldsymbol{\mu}, t)$ is independent from the brightness values of the image and the motion parameters at instant t. Let us call the above Jacobian $\mathbf{M}_0$.

With this approximation we have a constant Jacobian, that can be precomputed off-line, which is the best feature of the inverse compositional algorithm. At the same time, as it is additive with no factorisation, it could be aplied to any warping function.

9 Experimental Results

In order to compare the above algorithms, some experimental tests will be carried out. The image used to make the tests was generated synthetically using pov-ray[1]. Three algorithms have been implemented in MATLAB: forward compositional registration (see section 6), inverse compositional registration (see section 7) and approximate additive registration (see section 8).

For each algorithm we iteratively minimise the least-squares equations described in their corresponding sections and $\boldsymbol{\mu}$ is updated with $\delta\boldsymbol{\mu}$ until convergence is assured. This minimisation will stop when $\delta\boldsymbol{\mu}$ is bellow a certain threshold. We will compute at most 20 iterations of each algorithm. This will prevent us from computing too many iterations.

In this experiments we compare the performance of the three algorithms in terms of re-projection error, percentage of converged iterations, computation time and number of iterations.

The motion model, $\mathbf{y} = f(\mathbf{x}, \boldsymbol{\mu})$, that is used in this experiment represents the homography between a point in the original target region, $\mathbf{x}$, and its corresponding point onto image at instant t, $\mathbf{y}$. This homography can be parameterised using 8 components in the following form:

$$f(\mathbf{x}, \boldsymbol{\mu}) = \frac{1}{1 + \mu_7 x + \mu_8 y} \begin{bmatrix} (1 + \mu_1)x + \mu_3 y + \mu_5 \\ \mu_2 x + (1 + \mu_4)y + \mu_6 \end{bmatrix},$$

where $\mathbf{x} = (x, y)^\top$ is a pixel position in the target region, $\mathcal{R}$, and $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7, \mu_8)^\top$ is the vector of parameters of the homography.

We proceed as in [**?**]. We take the positions of the four corners of our square target region, $\mathcal{R}$, in the image. These positions will be perturbed randomly and the image will we warped according to the obtained quadrilateral. In this experiment σ will take values between 1 and 10. Then, we will compute the motion parameters, $\boldsymbol{\mu}$, that transform the initial target region into the distorted one. Figure 1 shows this procedure. For each σ we will average 100 different trials for each one of the three algorithms.

Comparison of the obtained results with ground truth is done by computing the Euclidean distance between the four corners of the deformed target region and those computed using the estimated $\boldsymbol{\mu}$. This distance is plotted against the noise variance in figure 2(a). It shows that, as σ increases, the forward compositional algorithm obtains a very poor result. When the perturbation level is 10, we get an average error of almost 30 pixels whereas this error is less than 3 pixels for the other two algorithms.

We also compute the frequency of convergence for each algorithm. We consider that an algorithm has converged if

[1] `www.povray.org`

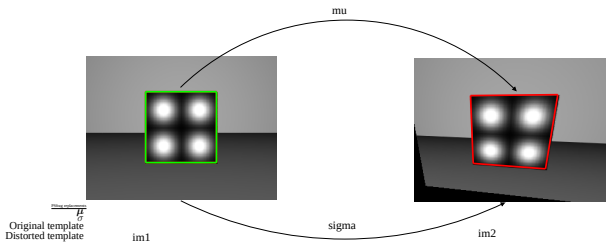
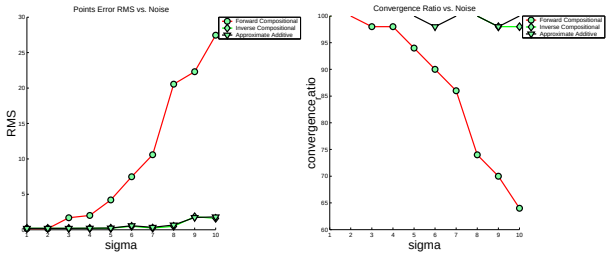


Figure 1: Initial setup for this experiment. The original target region is marked in green colour. This target region is distorted using Gaussian noise with standard deviation σ. The resulting target region is marked in red.

the error between the initial positions of the template corners and those computed using the computed parameters is less than 1.0 pixels. Results from these experiments are shown in figure 2(b). They show that the final motion parameters computed for the inverse compositional and approximate additive algorithms produced "correct" pixel positions in more than 95% of the cases. However, for the forward compositional algorithm the percentage decreases as σ increments its value, reaching a 65% of correct pixel positions for $\sigma = 10$. This is the explanation for the poor performance shown in figure 2(a)



(a) Points error RMS vs. noise for test image shown in fig .1.

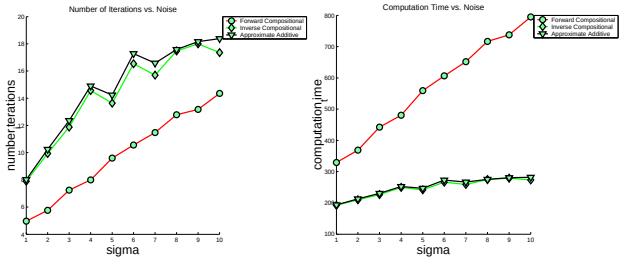
(b) Frequency of convergence vs. noise for test image shown in fig .1.

Figure 2: Average RMS of points error and frequency of convergence for test image shown in figure 1.

We also compute the number of iterations and the computation time. Figures 3(a) and 3(b) show, respectively, the average of the number of iterations and the computation time for each algorithm.Remember that the number of iterations can be at most of 20.

As we expected (see the figure 3(a)), the number of iterations increases with σ. Moreover, it is always smaller in the forward compositional case than in the other two algorithms. This means that the convergence of this algorithm is faster than the inverse compositional or the approximate additive one in terms of iterations. The computational cost of each iteration is bigger, though.

Finally, we examine the convergence rate of each algorithm. In order to do this, we plot the RMS of the error against the number of iterations (see figure 4).

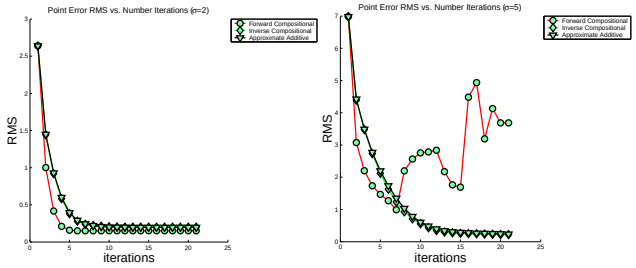


(a) Number of iterations vs. noise

(b) Computation time vs. noise for test image shown in fig .1.

Figure 3: Average iterations number and computation time for test image shown in figure 1.

Results plotted in these figures show that, although the forward compositional algorithm is the fastest and most accurate when it converges, it quickly degrades σ increases.



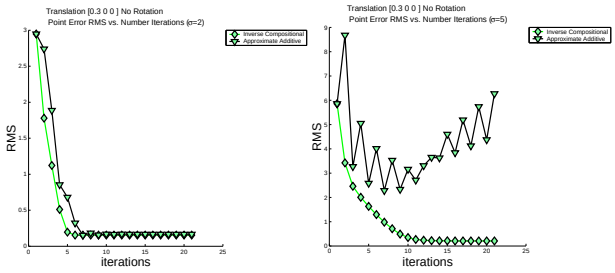
(a) Point Error RMS vs number of iterations for $\sigma = 2$.

(b) RMS vs number of iterations for $\sigma = 5$.

Figure 4: RMS of the error points vs. number of iterations for $\boldsymbol{\mu}_0$

The above results also show that the inverse compositional and approximate additive algorithm have similar performance. Note that the experiments are evaluated in the ideal conditions for the approximate additive algorithm; i.e. the Jacobian for our algorithm is evaluated at $\boldsymbol{\mu} = \boldsymbol{\mu}_0$.In this case the approximation in (30) is exact.

Let us evaluate both algorithms for a set of motion parameters different from $\boldsymbol{\mu}_0$. In this case, $\boldsymbol{\mu}$ represents a displacement of the template of 3 units in the X axis. Figure 5 shows the average values for 30 iterations of the RMS of the points error versus the number of iterations for $\sigma = 2$ and $\sigma = 5$. As we can see, results for approximate additive algorithm are poorer than for the inverse compositional one. This can be caused by an incorrect approximation of the Jacobian in (30).



(a) Point Error RMS vs number of iterations for $\sigma = 2$.

(b) RMS vs number of iterations for $\sigma = 5$.

Figure 5: RMS of the error points vs. number of iterations for $\boldsymbol{\mu}$ representing a translation of 3 units in x-axis.

10 Conclusions

Experiments suggest that the forward compositional algorithm performs the best in terms of accuracy and number of iterations. On the other hand, it lacks computational efficiency. The inverse compositional algorithm has similar performance to the forward compositional one but with better throughput. Unfortunately, all compositional algorithms together with the Hager and Belhumeur's one impose severe restrictions on their motion models.

On the other hand, incremental additive algorithms like the Lucas and Kanade's technique and the approximate additive registration do not impose such restrictive requirements on the motion model. However, Lucas and Kanade's algorithm lacks computational efficiency and the approximate additive algorithm does not converge properly when conditions are not ideal. Future lines of work toward improving the convergence of this algorithm are being explored in our research group.

Acknowledgements

The authors gratefuly acknowledge funding from the Spanish Ministry of Science and Technology under grant TIC2002-00591. E. Muñoz and J.M. Buensposada also acknowledge funding from the FPU program of the Spanish Ministry of Education and Culture.

Probabilistic models of image motion for recognition of dynamic content in video

G. Piriou, P. Bouthemy, N. Peyrard
IRISA / INRIA
Campus de Beaulieu
35042 Rennes cedex, France

J.-F. Yao
IRMAR & IRISA/INRIA
Campus de Beaulieu
35042 Rennes cedex, France

Abstract

We present new probabilistic motion models of interest for the detection of meaningful dynamic contents (or events) in videos. We separately handle the dominant image motion assumed to be due to the camera motion and the residual image motion related to scene motion. These two motion components are then represented by different probabilistic models which are further recombined for the event detection task. Two solutions are investigated for the residual motion. The motion models (both for camera motion and scene motion) associated to pre-identified classes of meaningful events are learned from a training set of video samples. The detection scheme proceeds in two steps which exploit different kinds of information and allow us to progressively select the video segments of interest using Maximum Likelihood (ML) criteria. The efficiency of the proposed approach is demonstrated on sport videos.

1 Introduction

One of the actual challenges in computer vision is to somehow approach the "semantic" content of video documents while dealing with physical image signals and numerical measurements. One objective can be to handle tasks such as video summarization, video retrieval or video surveillance. The main difficulty lies in the detection of "semantic concepts" from low-level features. The characteristics of a semantic event has to be expressed in terms of video primitives (color, texture, motion, shape ...) sufficiently discriminant w.r.t. content. This remains an open problem at the source of active research activities.

Different kinds of video features have already been considered in several approaches. In [1], the pixel chrominance components in the image are used to select keyframes maximally distinct and carrying the most information. [2] introduces statistical models for components of the video structure to classify video sequences into different genres (sports, news, movies, commercials, documentaries, ...). Recently, in [3], a semantic classification method based on SVM ("Support Vector Machine") using a motion pattern descriptor has been described.

The analysis of image motion is widely exploited for the segmentation of videos into meaningful units or for event recognition. Efficient motion characterization can be derived from the optical flow, as in [4] for human action change detection. In [5], the authors use very simple local spatio-temporal measurements, i.e., histograms of the spatial and temporal intensity gradients, to cluster temporal dynamic events. In [6], a principal component representation of activity parameters (such as translation, rotation ...) learned from a set of examples is introduced. The considered application was the recognition of particular human motions, assuming an initial segmentation of the body. In [7], two simple low-level motion features are used to characterize the activity level of video sequences.

In this paper, we propose new probabilistic motion models of particular interest for the detection of meaningful dynamic events. The motion information is captured through low-level motion measurements which convey more elaborated motion information than those used in [5], while still locally computable contrary to optic flow. They can be efficiently and reliabily computed in any video whatever its genre and its content. Our approach consists in handling separately the scene motion (i.e., the residual image motion) and the camera motion (i.e., the dominant image motion) in a sequence. Indeed, these two sources of motion bring important, different but complementary, information which have to be taken into account for event detection or classification.

We have investigated two different probabilistic motion models to specify the residual motion information. With the first model, we evaluate temporal cooccurrences of the local motion-related measurements which are first quantized. The resulting cooccurrence matrix is then viewed as a 2D histogram and represented by a 2D Gaussian mixture model. On the other hand, the second model directly exploits the local motion measurements, along with their temporal contrasts in order to capture the temporal motion evolution. Their histograms are computed over the video segment and are represented by a specific mixture model.

An original probabilistic model is also proposed to cope with the camera motion. It exploits 2D histograms of velocity vectors issued from the estimated affine motion models

accounting for the image dominant motion.
We apply this statistical framework to the detection of relevant events in a video following a two-step approach. The first step consists of a pre-selection of candidate segments among the successive segments of the processed video. It involves two pre-learned groups representing respectively "possibly important dynamic content" and " definitively not important dynamic content". The second step is a classification stage to recognize the relevant events (in terms of dynamic content) among the segments selected after the first step. Such a two-step process allows us to restrict the recognition issue on a limited and pertinent set of classes, to save computation time and to make the overall detection more robust and efficient.
The paper is organized as follows. In Section 2, we briefly present the motion measurements used. Section 3 describes the two statistical modelings of scene motion in a video that we have explored. Section 4 is dedicated to the probabilistic model for the camera motion. We present in Section 5 the full scheme for dynamic event detection. Experiments on sport videos are reported in Section 6 and Section 7 contains concluding remarks.

2 Motion Measurements

As stating above, we are investigating the probabilistic modeling of the motion content of a video. Such a modeling enables to derive a parsimonious motion representation while coping with errors in the motion measurements and with variability in a given kind of motion content. Furthermore, no analytical motion models are available to account for the variety of dynamic contents to be found in videos. We have to specify and learn them from the image data. Let us also stress that we aim at recognizing "broad" event classes and not particular "quantitative" motions. The proposed framework therefore exploits only low-level motion features for generality and efficiency purposes. Although the motion estimation step is not the purpose of this paper, we have first to briefly describe the motion measurements that we use.
It is possible to characterize the global image motion as proposed in [8], by computing at each pixel a local weighted mean of normal flow magnitude. However, the image motion is actually the sum of two motion sources: the dominant motion (usually assumed to be due to camera motion) and the residual motion (related to the scene motion). We believe that more information can be recovered when dealing with these two motions separately rather than only with the total motion. Thus, we compensate for the camera motion (more precisely, we cancel the estimated dominant image motion) in the sequence of images, in order to compute local motion-related measurements revealing only the residual image motion.
The image dominant motion is represented by a deterministic 2D affine motion model which is a usual choice:

$$\mathbf{w}_\theta(\mathbf{p}) = \begin{pmatrix} a_1 + a_2 x + a_3 y \\ a_4 + a_5 x + a_6 y \end{pmatrix}, \quad (1)$$

where $\theta = (a_i, i = 1, \ldots, 6)$ is the model parameter vector and $p = (x, y)$ is an image point. This simple motion model can handle different camera motions such as panning, zooming, tracking, (including of course static shots). Different methods are available to estimate such a motion model. We use the robust real-time multiresolution algorithm described in [9]. Let us point out that the motion model parameters are directly computed from the spatio-temporal derivatives of the intensity function.
Thus, the camera motion vector $\mathbf{w}_{\hat{\theta}_\mathbf{t}}(\mathbf{p})$ is available at each time t and for each pixel p. Then, the local motion-related measurement $v_{res}(p,t)$ is defined as the local mean of normal residual flows weighted by the square of the norm of the spatial intensity gradient. The normal residual flows are computed from the Displaced Frame Difference ($DFD_{\hat{\theta}_t}$) given by the estimated dominant motion. We finally get:

$$v_{res}(p,t) = \frac{\sum_{q \in \mathcal{F}(p)} \|\nabla I(q,t)\| . |DFD_{\hat{\theta}_t}(q)|}{N_{\mathcal{F}(p)} . \max\left(\eta^2, \sum_{q \in \mathcal{F}(p)} \|\nabla I(q,t)\|^2\right)}, \quad (2)$$

where $DFD_{\hat{\theta}_t}(q) = I(q + \mathbf{w}_{\hat{\theta}_\mathbf{t}}(\mathbf{q}), \mathbf{t} + \mathbf{1}) - \mathbf{I}(\mathbf{q}, \mathbf{t})$. $\mathcal{F}(p)$ is a local spatial window centered in pixel p and $N_{\mathcal{F}(p)}$ is the number of pixels in $\mathcal{F}(p)$. $\nabla I(q,t)$ is the spatial intensity gradient of pixel q at time t. η^2 is a predetermined constant related to the noise level. Such measurements have already been used for instance for the detection of independent moving objects in case of a mobile camera.
Figure 2 displays three images of an athletics TV program, the corresponding maps of dominant motion support and the corresponding maps of local motion-related measurements v_{res}.

3 Probabilistic models of scene motion

We have explored two types of statistics derived from the motion measurements defined in (2). The first one exploits temporal cooccurrences of the motion measurements which have to be first quantized. This choice enables to capture not only the motion magnitude but also the global temporal evolution of the motion magnitude. The computation of the cooccurrences being expensive in practice, we have defined an alternative approach where we directly consider the local motion-related measurements and their temporal gradients (contrasts). These two options for scene motion characterization and the respective designed probabilistic motion models are presented in the following two subsections.

3.1 GMM from temporal cooccurrences

With the first approach, the measures defined by (2) are quantized on a set Λ, so that for a video segment of length T and of spatial image support $\mathcal{R}$, the motion content is represented by the set of quantized local motion measurements $y = \{y(p,t), p \in \mathcal{R}, t = 1 \dots T\}$. The temporal cooccurrences distribution $\Gamma(y)$ of these quantities is a matrix $\{\Gamma(\nu,\nu'|y)\}_{(\nu,\nu')\in\Lambda^2}$ which is defined as follows:

$$\Gamma(\nu,\nu'|y) = \sum_{t=1}^{T-1} \sum_{p\in\mathcal{R}} \delta(\nu, y(p,t)) \cdot \delta(\nu', y(p,t+1)),$$

where $\delta(i,j)$ is the Kronecker symbol (equal to 1 if $i = j$ and to zero otherwise). The temporal cooccurrences matrix $\Gamma(y)$ is then considered as a 2D empirical histogram, and we model it by a 2D Gaussian mixture model (GMM). The log-likelihood of the sequence y is thus given by:

$$\ln P_{cooc.}(y) = \textstyle\sum_{(\nu,\nu')\in\Lambda^2} \Gamma(\nu,\nu'|y) \ln q(\nu,\nu') \quad (3)$$
$$\text{with } q(\nu,\nu') = \textstyle\sum_{k=1}^{K} \pi_k \phi(\nu,\nu'; m_k, \Sigma_k),$$

where K is the number of components in the mixture model and $\phi(\nu,\nu'; m_k, \Sigma_k)$ is the 2D Gaussian density function with mean vector m_k and covariance matrix Σ_k. The number of components K is determined with the Integrated Completed Likelihood criterion (ICL, [10]), and the Maximum Likelihood (ML) estimate of the model parameters is approximated using the Expectation-Maximisation algorithm.

3.2 DGMM from temporal contrasts

As an alternative to the computation of temporal cooccurrences, the temporal contrasts Δv_{res} of local motion-related measurements are considered. The contrasts are defined as the temporal difference of the variables v_{res} given by (2):

$$\Delta v_{res}(p,t) = v_{res}(p,t+1) - v_{res}(p,t). \quad (4)$$

We have computed the histograms of these expressions over different video segments and it has been found to be quite similar to a Gaussian distribution except a usually prominent peak at zero. Therefore, we model the temporal contrasts distribution by a specific mixture model with density:

$$P_{\Delta v_{res}}(\gamma) = \beta\delta_0(\gamma) + (1-\beta)\phi(\gamma; 0, \sigma^2)\mathbf{1}_{\gamma\neq 0} \quad (5)$$

where β is the mixture weight, δ_0 denotes the Dirac function at 0 ($\delta_0(\gamma) = 1$ if $\gamma = 0$ and $\delta_0(\gamma) = 0$ otherwise) and $\phi(\gamma; 0, \sigma^2)$ is the Gaussian density function with mean 0 and variance σ^2. The parameters β and σ^2 are estimated using the Maximum Likelihood criterion. Nevertheless, if we consider only the temporal contrasts Δv_{res}, the absolute motion magnitude would be lost. Consequently, the addition of the local motion-related measurements is required. They are also modeled by a mixture model of a Dirac function at 0 and a zero-mean Gaussian distribution, but the Gaussian distribution is here truncated to take into account only the positive values since by definition $v_{res}(p,t) \geq 0$. The mixture weight and the variance of the truncated Gaussian distribution are evaluated using the ML criterion. The global probabilistic residual motion model is then defined as the product of the two described models as follows :

$$P_{contr.} = P_{\Delta v_{res}}.P_{v_{res}} \quad (6)$$

Let us notice that in that case we do not need to quantize v_{res} and Δv_{res}, and we directly deal with the computed real values. Naturally, this model does not allow us to capture how the motion information is spatially distributed, but it is not necessary for the objective we consider here. In the sequel, this Dirac Gaussian mixture model will be referred to as DGMM.

4 Probabilistic model of camera motion

We have to design a probabilistic model of the camera motion to combine it with the probabilistic model of the residual motion in the recognition process. It could be possible to characterize directly the camera motion by the parameter vector θ defined in Section 2 and to represent its distribution over the sequence by a probabilistic model. The main difficulty in that case is to propose a valid probabilistic model. Indeed, if the distribution of the two translation parameters a_1 and a_4 can be easily described (these two parameters are likely to be constant within a video segment so that a Gaussian mixture could reasonably be used), the task becomes more difficult when dealing with the other parameters which are not constant anymore and which are not of the same nature. For this reason, we propose to build the map of the camera motion vectors obtained at each pixel of the image once the affine motion model is estimated, and to exploit these measurements as a 2D histogram. More precisely, at each time t, the motion parameters θ_t of the camera motion model (1) are estimated and the vectors $\mathbf{w}_{\hat{\theta}_t}(\mathbf{p})$ are computed for each point p of the image support. The values of the horizontal and vertical components of $\mathbf{w}_{\hat{\theta}_t}(\mathbf{p})$ are then finely quantized, and we form the empirical 2D histogram of their distribution over the considered video segment Finally, this histogram is represented by a mixture model of 2D Gaussian distributions. The number of components of the mixture and their parameters are estimated in a similar way as explained in subsection 3.1.

5 Event detection algorithm

We exploit now the designed probabilistic models of motion content for the task of event detection. We proceed in two

steps.
We suppose that the videos to be processed are segmented into homogeneous temporal units. This preliminary step is out of the scope of this paper which focuses on the motion modeling and recognition issues. To segment the video, we can use either a shot change detection technique or a motion-based video segmentation method. The first step of the event detection algorithm permits to sort the video segments in two groups, the first group contains the segments likely to contain the relevant events, the second one is formed by the video segments to be definitively discarded. Typically, if we consider sport videos, we try to first distinguish between "play" and "no play" segments. This step is based only on the residual motion which accounts for the scene motion, therefore only 1D models are used which saves computation. To this end, a motion model is learned off-line in a training stage for each group of segments. Then, the sorting consists in assigning the label "play" or "no play" to each segment of the processed video using the ML criterion. In practice, because of the large diversity of content in "play" or "no play" video segments in some videos, it can be useful to learn several models per group.
The second step of the proposed scheme consists in retrieving several specific events among the previously selected segments. Contrary to the first step, the two kinds of motion information (residual and camera motion) are required since the combination permits to characterize more precisely a specific event. For a given genre of video document, an off-line training step is again required. A residual motion model $\mathcal{M}^j_{res}$ (cooccurrence-based GMM or contrast-based DGMM) and a camera motion model $\mathcal{M}^j_{cam}$ have to be estimated from a given training set of video samples, for each type j of event to detect. Let $\{s_0, \ldots, s_N\}$ be the previously selected video segments. $\{z_0, \ldots, z_N\}$ are the corresponding motion measurements. If we consider the GMM approach described in subsection 3.1, $z_i = y_i$ where y_i are the quantized local motion-related measurements for segment s_i. If we consider the DGMM approach (subsection 3.2), $z_i = (\gamma_i, x_i)$ where γ_i are the temporal gradients of the local motion-related measurements x_i for segment s_i. The video segments retained after the first step are then labeled with one of the J learned models of dynamic events according to the Maximum Likelihood criterion. Thus, the label l_i of the segment s_i is defined as follows :

$$l_i = \arg \max_{j=1,\ldots,J} P_{\mathcal{M}^j_{res}}(z_i) \times P_{\mathcal{M}^j_{cam}}(w_i) \qquad (7)$$

where w_i represents the motion vectors corresponding to the estimated 2D affine motion models for the segment s_i, and $P_{\mathcal{M}^j_{res}}$ is either given by $P_{cooc.}$ (3) or by $P_{contr.}$ (6) according to the chosen option.

Figure 1: *Skating video:* Left to right and top to bottom: audience, scores, skating and different dance figures.

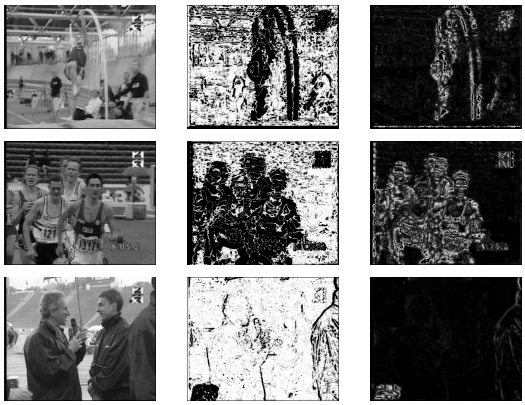


Figure 2: *Athletics video:* Three images at different time instants and their corresponding maps of dominant motion support (in white) and of local related-motion measurements v_{res} (zero-value in black). Top to bottom: pole vault, close-up of track race and interview.

6 Experimental results

In subsection 6.1, we give the results of the first step of the designed method for two different sport programs. Results of event detection are shown and commented in subsection 6.2. Finally, we report experimental comparisons in subsection 6.3. We have carried out experiments on several video programs. Due to page limitation, we report here results obtained on two different sport videos.

6.1 Selecting video segments

The first video is a figure skating (dance) TV program. We want to distinguish between "play" segments which correspond to skating (simple skating motion, artistic effects, dance movements) and "no play" segments involving low-

	P	R
GMM	1	0.83
DGMM	0.95	0.90

Table 1: *Skating video:* Results of the first step of the event detection algorithm for the two considered models (GMM and DGMM). $P =$ precision rate and $R =$ recall rate.

	P	R
GMM	0.79	0.94
DGMM	0.84	0.94

Table 2: *Athletics video:* Results of the first step of the event detection algorithm for the two considered models (GMM and DGMM). P = precision rate and R = recall rate.

level activity (views of the audience, static shot like waving at the beginning and the end of each show, skaters waiting for the scores) as illustrated in Figure 1. The first 23 minutes of the video (two shows) are used as the training set and the last 9 minutes (one show) form the test set. The video segments of the test set are sorted as described in section 5. Here, each group ("play", "no play") is represented by several motion models; then the ML criterion involves to maximise also over the different models of each group. The precision rate P and the recall rate R are defined as follows:

$$P = \frac{\#\text{correct}}{\#\text{correct} + \#\text{intrusive}} \text{ and } R = \frac{\#\text{correct}}{\#\text{correct} + \#\text{missed}},$$

where $\#$correct is the number of video segments labeled as "play" segments and which effectively belong to this group, $\#$intrusive is the number of "no play" segments labeled as "play" segments and $\#$missed is the number of "play" segments labeled as "no play" segments. Table 1 contains the first-step results obtained respectively with the GMM and DGMM models for the residual motion. Results are comparable and quite satisfactory.

We have also processed one athletics TV program which is formed by 25500 images. The training set is 10 minutes long and the test set is 7 minutes long. The "play" segments are formed by jump events and track race shots and the "no play" segments contain interview shots and large views of the stadium. Some representative images of this video are displayed on Figure 2. We present in Table 2 the first-step results obtained on that video. Again, satisfactory results are obtained.

6.2 Detecting relevant events

The aim is now to detect the relevant events of the athletics video among the segments selected as "play" segments (16 segments selected from the first step when using the GMM model and 17 when using the DGMM model for the residual

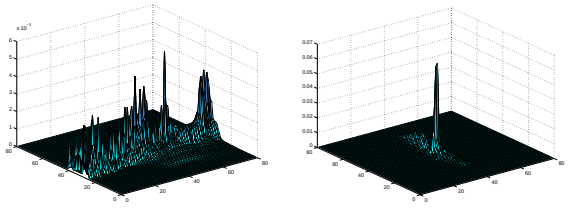


Figure 3: *Athletics video:* 2D histograms of the estimated camera motion vectors. Left: pole vault, right: wide-shot of track race.

	Assigned label			
	Pv	Pr	Rw	Rc
Pv	2	0	0	0
Pr	0	2	0	0
Rw	0	1	3	1
Rc	0	0	0	6
NP	0	1	0	0

	Assigned label			
	Pv	Pr	Rw	Rc
Pv	2	0	0	0
Pr	0	2	0	0
Rw	0	1	4	1
Rc	0	0	0	6
NP	0	1	0	0

Table 3: *Athletics video:* Classification matrix obtained with the two-step event detection method. Left: with the GMM model. Right: with the DGMM model.

Figure 4: *Athletics video:* Top: ground-truth, middle: results obtained with GMM model, bottom: results obtained with DGMM model. Grey: "no play", red: pole vault, yellow: replay of pole vault, green: wide-shot of track race, blue: close-up of track-race

motion). For this second step, we introduce the probabilistic camera motion model. The 2D histograms of the estimated camera motion vectors for different classes are plotted on Figure 3. The four events we try to detect are the following: pole vault (*Pv*), replay of pole vault (*Pr*), wide-shots of track race (*Rw*) and close-up of track race (*Rc*). Let us point out that the class "Replay of pole vault" contains the run-up and the jump, whereas the class "pole vault" contains only the jump. On Figure 4, the processed video is represented by a time-line exhibiting the duration of video segments. Figure 4 represents in a combined way the results of the two-step event detection method, also reported in a different and separate way by Table 2 (first step) and Table 3 (second step). "No play" is displayed in grey and a color is associated to each event class. The first row represents the ground-truth. The second and the third ones show the results obtained respectively using GMM model and DGMM model as for the residual motion. The detection also involves the camera motion model. From Figure 4 and Table 2 and 3, we can infer that the majority of events are appropriately detected. Let us note that the intrusive segments appear on the line *NP* in Table 3. The classification errors concern two segments (two short segments at the end of the video sequence) belonging to the class "wide-shot of track race". The misclassification is due to the fact that the first segment involves a scene which is between wide-angle shot and close-up, and the second one is quite similar to the run-up of pole vault in terms of movement.

6.3 Experimental comparisons

Table 4 contains results when considering the camera motion only. Conversely, Table 5 gives results obtained when using the residual motion model only. These two tables

	Assigned label			
	Pv	Pr	Rw	Rc
Pv	2	0	0	0
Pr	0	2	0	0
Rw	0	1	2	2
Rc	1	0	0	5
NP	0	1	0	0

	Assigned label			
	Pv	Pr	Rw	Rc
Pv	2	0	0	0
Pr	0	2	0	0
Rw	0	1	1	4
Rc	1	0	0	5
NP	0	1	0	0

Table 4: *Athletics video:* Classification matrix obtained when considering the camera motion only. Left: on the segments selected after the first step based on the GMM model. Right: on the segments selected after the first step based on the DGMM model.

	Assigned label			
	Pv	Pr	Rw	Rc
Pv	2	0	0	0
Pr	0	2	0	0
Rw	0	0	5	0
Rc	0	0	2	4
NP	0	0	1	0

	Assigned label			
	Pv	Pr	Rw	Rc
Pv	2	0	0	0
Pr	0	2	0	0
Rw	2	3	1	0
Rc	2	0	0	4
NP	0	0	1	0

Table 5: *Athletics video:* Classification matrix obtained when considering the residual motion only. Left: with the GMM model. Right: with the DGMM model.

demonstrate that the combination of both motions (residual motion and camera motion) yields better results as for event detection.

On the other hand, whereas the GMM model was supposed to capture more information with the cooccurences statistics, Tables 2 and 3 show that the DGMM model finally yields similar results while being less time consuming. Indeed, for the GMM model the computation time is 0.8 sec/image with a Pentium IV 2.4 Ghz, while it is 0.2 sec/image for the DGMM model.

7 Conclusion

In this paper, we have introduced new probabilistic motion models which can be easily learned and computed from the image data and can handle a large variety of dynamic video contents. We explicitly handle the information related respectively to the scene motion and to the camera motion. Two probabilistic models involving different statistical representations of the scene motion have been investigated and compared. We have also introduced an original probabilistic modeling of camera motion. These motion models were proven to be efficient and appropriate for event detection in videos. The proposed method induces a low computation time, and accurate results on sport videos have been reported.

The proposed two-step method for event detection is general and does not exploit very specific knowledge (related to the video genre, e.g., type of sport) and dedicated solutions. It can thus be successfully applied to a large range of videos. In the same time, due to the considered statistical framework, it is flexible enough to properly introduce prior on the classes (then, skipping to MAP instead of ML criterion) if available, or to incorporate other useful information. A complete event detection scheme should also integrate colour (the dominant colour is useful for instance to account for the presence of the play field or the tennis court in sport videos), or audio features which are also of obvious interest when processing videos. Such developments are indeed in progress for a video summarization application.

Acknowledgments

This work has been partly supported by Brittany Council (Ph-D thesis grant) and by the IST project LAVA. The authors would like to thank INA, Direction de la Recherche, for providing the videos.

References

[1] J. Vermaak, P. Pérez, and M. Gangnet, "Rapid summarization and browsing of video sequences," *British Machine Vision Conference, Cardiff*, September 2002.

[2] N. Vasconcelos and A. Lippman, "Statistical models of video structure for content analysis and characterization," *IEEE Trans. on Image Processing*, vol. 9, no. 1, pp. 3–19, Jan. 2000.

[3] Y-F. Ma and H-J. Zhang, "Motion pattern-based video classification retrieval," *EURASIP Journal on Applied Signal Processing*, vol. 2, pp. 199–208, March 2003.

[4] Y. Rui and P. Anandan, "Segmenting visual actions based on spatio-temporal motion patterns," *IEEE Int. Conf. on Computer Vision and Pattern Recognition, Hilton Head, SC*, vol. 1, pp. 111–118, 2000.

[5] L. Zelnik-Manor and M. Irani, "Event-based video analysis," *IEEE Int. Conf. on CVPR, Kauai, Hawaii*, vol. 2, pp. 123–130, December 2001.

[6] Y. Yacoob and J. Black, "Parametrized modeling and recognition of activities," *Sixth IEEE Int. Conf. on Computer Vision, Bombay, India*, pp. 120–127, 1998.

[7] K. A. Peker, A. A. Alatan, and A. N. Akansu, "Low-level motion activity features for semantic characterization of video," *IEEE Int. Conf. on Mulimedia and Expo, ICME'00*, 2000.

[8] R. Fablet, P. Bouthemy, and P. Pérez, "Non-parametric motion characterization using causal probabilistic models for video indexing and retrieval," *IEEE Trans. on Image Processing*, vol. 11, no. 4, pp. 393–407, 2002.

[9] J-M. Odobez and P. Bouthemy, "Robust multiresolution estimation of parametric motion models," *J. of Visual Communication and Image Representation*, vol. 6, no. 4, pp. 348–365, December 1995.

[10] C. Biernacki, G. Celeux, and G. Govaert, "Assessing a mixture model for clustering with the Integrated Completed Likelihood," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 3, pp. 719–725, 2000.

A Nonparametric Estimation Method for Poisson Noise Reduction and Structure Preserving in Confocal Microscopy

Charles Kervrann
IRISA / INRA - BIA
Campus de Beaulieu
35042 Rennes Cedex, France

Abstract

In domains like confocal microscopy, the imaging process is based on detection of photons. It is established the additive Gaussian noise model is a poor description of the actual photon-limited image recording, compared with that of a Poisson process. This motivates the use of restoration methods optimized for Poisson noise distorted images. In this paper, a novel restoration approach is proposed for Poisson noise reduction and discontinuities preservation in images. The method is based on a local modeling of the image, with an adaptive choice of a neighborhood (window) around each pixel in which the applied model fits the data well. The restoration technique associates with each pixel the weighted sum of data points within the window. We describe a statistical method for choosing the optimal window size, in a manner that varies at each pixel, with an adaptive choice of weights for every pair of pixels in the window. It is worth noting the proposed technique applied to confocal microscopy is data-driven and does not require the hand tuning of parameters.

1 Introduction

Confocal systems offer the chance to image thick biological tissue in 2D or 3D dimensions. They operate in the bright-field and fluorescence modes, allowing the formation of high-resolution images. At each point of the image, the emitted fluorescence for the object is focused on the detector. This light is converted by a photomultiplier tube (PMT) into a an electrical signal and represented by a discrete value after an A/D conversion [12]. Finally, the PMT essentially behaves as a photon counter and the distortions caused by the quantum nature of the photon detection process are better described by Poisson statistics. Operations that invert these distortions of the microscope are then necessary to improve the quantitative analysis of images. Generally restoration methods yield an estimate of the original image given an imaging model, a noise model and additional criteria. In previous work [15, 18, 19], the Richardson-Lucy algorithm, which is a maximum likelihood estimator for the intensity of a Poisson process, incorporates a non-negativity constraint in the algorithm. This algorithm is generally capable of partially reducing the distortions found in confocal 3D images. However, it is sensitive to noise [18] and additional methods (Gaussian prefiltering) are necessary to produce better restoration results [19].

In this paper, we also address the adaptive image restoration problem and present a nonparametric estimation method that smooth homogeneous regions and inhibits smoothing in the neighborhood of discontinuities. The observed data are imperfect and in the form of Poisson process. Since we do not address the image formation of the confocal fluorescence microscope, ideally modeled as a convolution of the object function with the point spread function [12], the proposed method can be seen also as a sophisticated prefiltering method before starting the more complex *deconvolution* process using the Lucy-Richardson algorithm [19]. The proposed *adaptive window approach* is conceptually very simple being based on the key idea of estimating a locally regression function with an adaptive choice of the window size (neighborhood) for which the applied model fits the data well [14]. At each pixel, we estimate the regression function by iteratively growing a window and adaptively weighting input data to achieve an optimal compromise between the bias and variance [9, 8, 10]. The proposed algorithm complexity is actually controlled by simply restricting the size of the larger window and setting the window growing factor. In contrast to most digital diffusion-based filtering processes for which the input noisy image is “abandoned” after the first iteration [13, 3], the adaptive window approach recycles at each step the original data. Other related works to our approach are nonlinear Gaussian filters [7] and their recent evolutions [17, 1]) that essentially average values within a local window, but changes the weights according to local differences in the intensity [17, 1].

2 A nonparametric approach

In photon-limited imaging, the major source of errors is Poisson noise due to the discrete nature of photon detection. Unlike Gaussian noise, Poisson noise is signal dependent, which makes separating signal from noise a very difficult task [11]. However, by applying the Anscombe transform [16], the Poisson data $f_i \sim \text{Poiss}(\nu_i)$ of intensity ν_i are transformed to data with a Gaussian distribution with variance $\sigma^2 = 1$. Such an assertion is asymptotically correct as $\nu_i \to \infty$. The Anscombe transform is given by:

$$Y_i \;=\; 2\sqrt{f_i + \frac{3}{8}}. \tag{1}$$

This transformation allows one to use well-studied methods for Gaussian noise on data corrupted with the much trickier Poisson noise. Then we applied a method developed for Gaussian noise. After denoising, the inverse Anscombe transform is applied.

2.1 Image model and basic idea

We observe the regression function u with some additive error ε_i: $Y_i = u(X_i) + \varepsilon_i, \quad i = 1, \cdots, n$, were $X_i \in \mathbb{R}^d, d = 2, 3$, represents the spatial coordinates of the discrete image domain S of n pixels and $Y_i \in \mathbb{R}$ is the observed intensity at location X_i. We suppose the errors ε_i to be independent identically distributed zero-mean random variables with unknown variances, i.e., $\text{var}(\varepsilon_i) = \sigma^2$. However, to cope with more complex degradations, the noise variance is also an unknown parameter of the method.

A classical nonparametric estimation approach is based on the structural assumption that regression function $u(x)$ is constant in the vicinity of a point x. An important question under such an approach is first how to determine for each pixel the size and shape of the neighborhood under concern from image data. The regression function $u(x)$ can be then estimated from the observations lying in the estimated neighborhood of x by a local maximum likelihood (ML) method.

Our procedure is iterative and mostly realizes this idea. For the first step, suppose we are given a local window $U_i^{(0)}$ containing the point of estimation X_i. By $\hat{u}_i^{(0)}$ we denote an approximation of $\hat{u}^{(0)}(X_i)$. We can calculate an initial ML estimate $\hat{u}_i^{(0)}$ at point X_i (and its variance $\hat{\vartheta}_i^{(0)}$) by averaging observations over a small neighborhood $U_i^{(0)}$ of X_i as

$$\hat{u}_i^{(0)} = \frac{1}{|U_i^{(0)}|} \sum_{X_j \in U_i^{(0)}} Y_j \quad \text{and} \quad \hat{\vartheta}_i^{(0)} = \frac{\hat{\sigma}^2}{|U_i^{(0)}|} \tag{2}$$

where $\hat{\sigma}^2$ is the unknown estimate of the noise variance and $|U_i^{(0)}|$ denotes the number of points $X_j \in U_i^{(0)}$. At the next iteration, a larger neighborhood $U_i^{(1)}$ with $U_i^{(0)} \subset U_i^{(1)}$ centered at X_i is considered, and every point X_j from $U_i^{(1)}$ gets a weight $w_{ij}^{(1)}$ which is defined by comparing the estimates $\hat{u}_i^{(0)}$ and $\hat{u}_j^{(0)}$ obtained at the first iteration. Then we recalculate the estimate $\hat{u}_i^{(1)}$ as the weighted average of data points lying in the neighborhood $U_i^{(1)}$. We continue this way, increasing with k the considered neighborhood $U_i^{(k)}$; for each $k \geq 1$, the ML estimate $\hat{u}_i^{(k)}$ and its variance are given by

$$\hat{u}_i^{(k)} = \sum_{X_j \in U_i^{(k)}} w_{ij}^{(k)}\, Y_j \text{ and } \hat{\vartheta}_i^{(k)} = \hat{\sigma}^2 \sum_{X_j \in U_i^{(k)}} \left[w_{ij}^{(k)}\right]^2 \tag{3}$$

where weights $w_{ij}^{(k)}$ are continuous variables ($0 \leq w_{ij}^{(k)} \leq 1$), computed by comparison of the preceding estimates $\hat{u}_i^{(k-1)}$ and $\hat{u}_j^{(k-1)}$. In the next section, statistical arguments for calculating weights $w_{ij}^{(k)}$ are given. To stabilize the procedure, we also add a control step to valid the window size for each pixel, by comparing the new estimate $\hat{u}_i^{(k)}$ with the estimate $\hat{u}_i^{(k-1)}$ obtained at the preceding iteration [14]. In equation (3), the weight function $w_{ij}^{(k)}$ does not depend on input data but are only calculated from neighboring local estimates, which contributes to the regularization effect. Finally, a global convergence criterion is introduced to optimally stop the estimation procedure.

2.2 Adaptive weights

In our approach, we may decide on the basis of the estimates $\hat{u}_i^{(k-1)}$ and $\hat{u}_j^{(k-1)}$, whether points X_i and $X_j \in U_i^{(k)}$ are in the same region or not and then prevent from significant discontinuities oversmoothing [14]. In the local Gaussian case, significance is measured using a contrast $|\hat{u}_i^{(k-1)} - \hat{u}_j^{(k-1)}|$. If this contrast is high compared to the local variance $\hat{\vartheta}_i^{(k-1)}$, then X_j should not participate to the estimation of $\hat{u}_i^{(k)}$ and $w_{ij}^{(k)} \to 0$. Hence, motivated by the robustness and smoothing properties of the Huber M-estimator in the probabilistic approach of image denoising [2], we introduce the following related weight function (but other weight functions are possible [3]):

$$w_{ij}^{(k)} = \frac{g_{ij}^{(k)}}{\displaystyle\sum_{X_j \in U_i^{(k)}} g_{ij}^{(k)}}, \tag{4}$$

$$g_{ij}^{(k)} = \begin{cases} 1 & \text{if } \left|\hat{u}_i^{(k-1)} - \hat{u}_j^{(k-1)}\right| \leq 3\sqrt{\hat{\vartheta}_i^{(k-1)}} \\ \dfrac{3\sqrt{\hat{\vartheta}_i^{(k-1)}}}{\sqrt{\left|\hat{u}_i^{(k-1)} - \hat{u}_j^{(k-1)}\right|^2 + \epsilon}} & \text{otherwise} \end{cases}$$

with ϵ positive and sufficiently small (of the order 10^{-4}). Here $3\sqrt{\hat{\vartheta}_i^{(k-1)}}$ is related to the spatially varying fraction

of contamination of the Gaussian distribution (*"rule of 3 sigma"* for Gaussian distributions.): for the majority of points $X_j \in U_i$, the differences $\hat{u}_i^{(k-1)} - \hat{u}_j^{(k-1)}$ can be approximatively modeled as being constant (zero) with random Gaussian noise. Large differences are assumed to be outliers which should not have a large effect on the estimator.

2.3 Localization by a window

Statistical inference under such a structural assumption focuses on searching for every point X_i the largest neighborhood (window) U_i where the hypothesis of structural homogeneity is not rejected. The classical measure of the closeness of the estimator $\hat{u}$ obtained in the window U_i to its target value u is the mean squared error (MSE) which is decomposed into the sum of the squared bias $[\text{Bias}\,(\hat{u}_i)]^2$ and variance $\hat{\vartheta}_i$. As explained before, we should choose a window that achieves an optimal compromise between the squared bias and variance. Accordingly, we make the reasonable assumption that the squared bias is an increasing function of the neighborhood size and the variance is a decreasing function of the neighborhood size. Then, in order to minimize the MSE we search for the window where the squared bias and the variance of the estimate are equal. The corresponding critical MSE is ($E[\cdot]$ denotes the mathematical expectation):

$$\text{MSE}\left(\hat{u}_i^{(k^\star)}\right) \;=\; \left[\text{Bias}\left(\hat{u}_i^{(k^\star)}\right)\right]^2 + \hat{\vartheta}_i^{(k^\star)} \;=\; 2\hat{\vartheta}_i^{(k^\star)}. \quad (5)$$

Now, let us introduce a finite set of k_M windows $\{U_i^{(0)}, \cdots, U_i^{(k_M)}\}$ centered at $X_i \in S$, with $U_i^{(k)} \subset U_i^{(k+1)}$, starting with a small $U_i^{(0)}$ and the corresponding estimates $\hat{u}_i^{(0)}$ of the true image $u(X_i)$. Denote by $U_i^{(k^*)}$ the ideal window size corresponding to the minimum value of the MSE. Then $U_i^{(k^*)}$ gives the optimal bias-variance tradeoff and the optimal window $U_i^{(k^*)}$ can be obtained according to the following statistical rule [8, 9, 10]:

$$k^* = \max\left\{k : \forall k' < k : \left|\hat{u}_i^{(k)} - \hat{u}_i^{(k')}\right|^2 \leq 8\hat{\vartheta}_i^{(k')}\right\}. \quad (6)$$

In other words, as long as successive estimates $\hat{u}_i^{(k)}$ stay close to each other, we decide that the bias is small and the size of the estimation window can be increased to improve the estimation of the constant model (and to decrease the variance of the estimate $\hat{u}_i^{(k)}$). If an estimated point $\hat{u}_i^{(k')}$ appears far from the previous ones, we interpret this as the dominance of the bias over the variance term. For each pixel, the detection of this transition enables to determine the critical window size that balances bias and variance.

2.4 Stopping rule

The stopping rule can be used to save computation time if two successive solutions are very close and prevents from an useless setting of the larger window size. In our approach, we adopt the so-called Csiszár's I-divergence [5, 19] to detect global convergence defined as:

$$I(\hat{u}^{(k)}, \hat{u}^{(k+1)}) = \sum_{i=1}^{n}\left[\hat{u}_i^{(k)} \log \frac{\hat{u}_i^{(k)}}{\hat{u}_i^{(k+1)}} - \hat{u}_i^{(k)} + \hat{u}_i^{(k+1)}\right]. \quad (7)$$

In practice, the I-divergence is normalized with its maximal occurring value at iteration $k = 0$. When $I(\hat{u}^{(k)}, \hat{u}^{(k+1)})$ sinks under a threshold (of the order 10^{-3} for typical images) that sufficiently accounts for convergence, the algorithm is stopped at the final iteration $k_c = k$, with $k_c \leq k_M$, where k_M is the maximal value for k set by the user. Finally, the windows size increases at each iteration k if the convergence criterion is not met (or $k \leq k_M$), but the estimate $\hat{u}_i^{(k)}$ at iteration k is frozen.

3 Algorithmic procedure

3.1 The procedure

The key ingredient of the procedure is an increasing sequence of neighborhoods $U_i^{(k)}, k = 0, 1, \cdots, k_M$ with $U_i^{(k)} \subset U_i^{(k+1)}$ centered at each image pixel X_i. In what follows, $|U_i^{(k)}|$ denotes the number of points X_j in $U_i^{(k)}$, i.e. $|U_i^{(k)}| = \#\{X_j \in U_i^{(k)}\}$. The procedure is as follows.

Initialization For each point X_i, we calculate initial estimates $\hat{u}_i^{(0)}$ and $\hat{\vartheta}_i^{(0)}$ using equation (2) and set $k = 1$. Here $\hat{\sigma}^2$ is the noise variance robustly estimated from data as it is explained in Section 3.2.

Adaptation For all X_j in $U_i^{(k)}$, we compute weights $w_{ij}^{(k)}$ using equation (4) and new estimates $\hat{u}_i^{(k)}$ and $\hat{\vartheta}_i^{(k)}$ using equation (3).

Control After the estimate $\hat{u}_i^{(k)}$ has been computed, we compare it to the previous estimates $\hat{u}_i^{(k')}$ at the same point X_i for all $k' < k$. From the rule (6), if there is at least one index $k' < k$ such that $\left|\hat{u}_i^{(k)} - \hat{u}_i^{(k')}\right| > 2\sqrt{2}\,\sqrt{\hat{\vartheta}_i^{(k')}}$ then we do not accept $\hat{u}_i^{(k)}$ and keep the estimates $\hat{u}_i^{(k-1)}$ from the preceding iteration as the final estimate at location X_i [8, 10]. This estimate is unchanged if $k > k^*$.

Stopping Stop the procedure if $k = k_M$ or if $I(\hat{u}^{(k)}, \hat{u}^{(k+1)}) < 10^{-3}$, otherwise increase k by 1 and continue with the adaptation step.

3.2 Implementation details

In this section, we discuss how the parameters of the procedure can be obtained.

The window $U^{(0)}$ should be taken possibly small. We naturally choose $|U^{(0)}| = 1$. The sequence of neighborhoods $\{U^{(k)}\}$ is more or less pragmatical and not supported by theory. The sequence should satisfy the two following conditions: $X_i \in U_i^{(0)}$ and $U_i^{(k-1)} \subset U_i^{(k)}$. It can be recommended to select sequences $\{U_i^{(k)}\}$ in a way that the number $|U_i^{(k)}|$ of points in every such neighborhood grows exponentially with k. In our experiments, we arbitrarily use neighborhoods $U^{(k)}$ corresponding to successive square windows of size $|U^{(k)}| = (2k+1) \times (2k+1)$ pixels with $k = 0, 1, 2, \ldots, k_M$.

We can use the parameter k_M to bound the numerical complexity of the procedure. However, increasing k_M allows additional variance reduction in large homogeneous regions but usually does not change the estimates where local structure is present. In our experiments, $k_M = 15$ satisfies a good compromise and over-estimates the number of necessary iterations. In addition, the I-divergence criterion prevents from a precise tuning of this parameter.

The estimation procedure described earlier relies on the preliminary and off-line estimation of the noise variance. In most applications, the noise variance σ^2 is unknown and an estimate $\hat{\sigma}^2$ can be obtained from data as

$$\hat{\sigma}^2 \;=\; \frac{1}{|S|} \sum_{X_j \in S} \hat{e}_j^2 \tag{8}$$

where S denotes the rectangular image domain and pseudo-residuals $\hat{e}_j$ can be computed as (we note Y_{j_1,j_2} the observation Y_j at site $j = (j_1, j_2)$) [6]:

$$\hat{e}_j = \frac{4Y_{j_1,j_2} - (Y_{j_1+1,j_2} + Y_{j_1-1,j_2} + Y_{j_1,j_2+1} + Y_{j_1,j_2-1})}{\sqrt{20}}. \tag{9}$$

In presence of discontinuities, an estimate of the noise variance based on robust statistics is preferable. In this framework, high discontinuities correspond to statistical outliers with respect to local image contrasts. As in [3], we suggest to define $\hat{\sigma}^2$ as the following robust estimate:

$$\hat{\sigma} \;=\; 1.4826 \text{ median } (|\,|\hat{e}_S| - \text{median}|\hat{e}_S|\,|) \tag{10}$$

where $\hat{e}_S = \{\hat{e}_1, \hat{e}_2, \cdots, \hat{e}_n\}$ is the set of $n = |S|$ pseudo-residuals of the entire image.

3.3 Properties

We study an idealized situation where the underlying image is constant: $u(x) = u_o$. In this situation, we can show that all estimates $\hat{u}_i^{(k^\star)}$ coincide with the mean values of observations Y_i with a very large probability, and the deviations $\hat{u}_i - u_0$ are of the order $n^{-1/2}$. Additionally, if the contrast of the image is sufficiently large compared with the level of noise, we typically obtain $w_{ij}^{(k)} = 0$ for all pairs (X_i, X_j) in two distinct regions with a probability close to 1. Finally, since $\hat{u}_i^{(k)}$ is restricted to a local neighborhood $U_i^{(k)}$, the estimation procedure does not yield a segmentation of the image domain. But, if the noise is small compared to the contrast of the image, then, for sufficiently large $k^\star$, we have $\hat{u}_i^{(k)} \simeq \hat{u}_j^{(k)}$.

The complexity of the whole procedure is of the order $n(|U^{(1)}| + |U^{(2)}| + \cdots + |U^{(k_c)}|)$ if an image contains n pixels and $k_c \leq k_M$. Since $|U^{(k)}|$ grows exponentially in our set-up, the whole complexity is of order $O(n|U^{(k_c)}|)$.

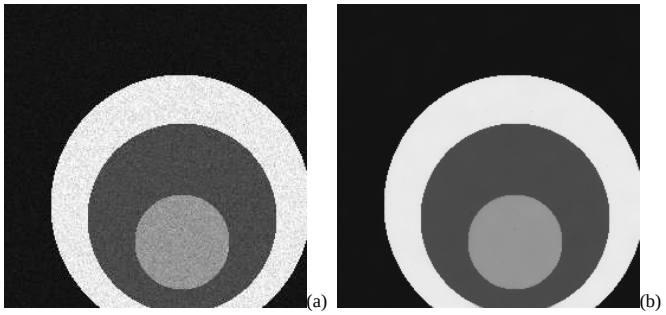


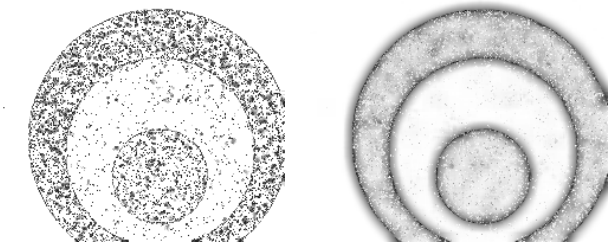
Figure 1: Corrupted synthetic image with Poisson noise (a) and denoised image (b).

4 Experiments

The potential of the adaptive window method is first shown on a synthetic image artificially corrupted with a Poisson noise (Fig. 1a, PSNR = 28.8 db). In Fig. 1b, the Poisson noise is drastically reduced in a natural manner and significant geometric features such as object boundaries, and original contrasts are visually well preserved (PSNR = 40.1 db). In this experiment, it is confirmed the additive Gaussian noise is a poor description of the actual photon-limited image recording (PSNR = 39.6). In addition, Figs. 2a (Poisson statistics) and 2c (Gaussian statistics) show the locations and sizes of optimal estimation windows; we have coded small windows with black and large windows with white. As expected, small windows are in the neighborhood of image gradients in Fig. 2a but are more dispersed in regions with high intensity values (Fig. 2c, Gaussian statistics). Finally, Figs. 2b and 2d show respectively the corresponding sets of average weights $\{\overline{w_{ij}^{(k^\star)}}\}$ defined as $\overline{w_{ij}^{(k^\star)}} = |U_i^{(k^\star)}|^{-1} \sum_{X_j \in U_i^{(k^\star)}} w_{ij}^{(k^\star)}$, where dark values indicate the presence of high image discontinuities. Note the average weights are notably more diffuse in Fig. 2d

(a) locations/sizes of windows (Poisson statistics)

(b) image of average weights $\overline{w_{ij}^{(k^\star)}}$ (Poisson statistics)



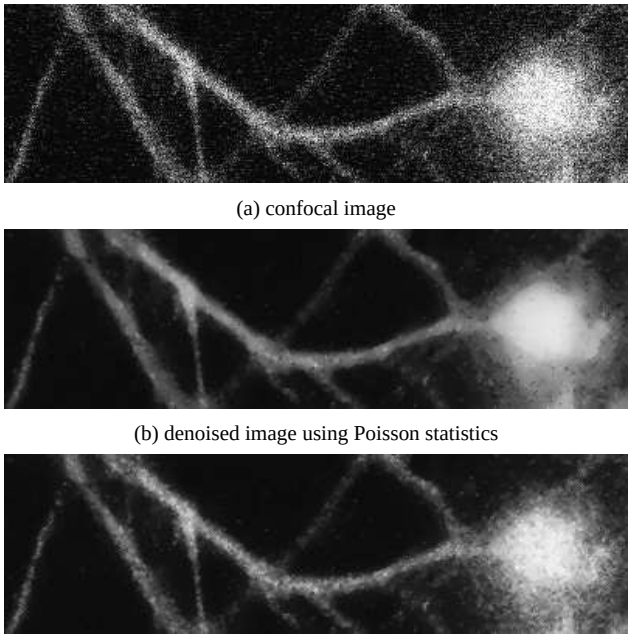
(c) locations/sizes of windows (Gaussian statistics)

(d) image of average weights $\overline{w_{ij}^{(k^\star)}}$ (Gaussian statistics)

Figure 2: Visualization of estimation windows and weights.

(Gaussian statistics). In this experiment, the final iteration $k_c = 7$ was determined autonomously by the I-divergence criterion ($k_M = 15$). The processing of the 256×256 image required typically 3 seconds on a PC (2.6 Ghz, Pentium IV) using a standard C++ implementation of the algorithm.

In the subsequent experiments, we have tested the algorithm (also implemented for processing 3D data) on confocal fluorescence microscopy images that contain complex structures. Some of the current applications in biological studies are in neuron research. The biological study aims at examining the effect of intracellular calcium concentration (Ca^2) on neurite outgrowth in individual neuronal cells. Indeed, Ca^2 is known to be implicated as an important regulator of neurite extension. We have tested the proposed denoising algorithm on a 2D confocal microscopy 115×512 image (Fig. 3, courtesy of INSERM 413 IFRMP n^o23, Rouen, France) depicting a neurite in cultured cerebellar granule cells. Fig. 3a shows the highly noisy image where high gray-level values correspond to elevated calcium concentration. Figures 3b and 3c contain the denoised images using the adaptive window method combined with respectively Poisson statistics and Gaussian statistics; the image denoised using Poisson statistics contains larger homogeneous regions. The adaptive window method using Poisson statistics has been also applied to 3D data: a typical 2D image taken from a 3D stack of 20 images depicting membranes of about fifty cultured human cells is shown in Fig. 4 (courtesy of INRA - UFDNH, Nantes, France). The image is denoised using the set of parameters used in the previous experiments.



(a) confocal image

(b) denoised image using Poisson statistics

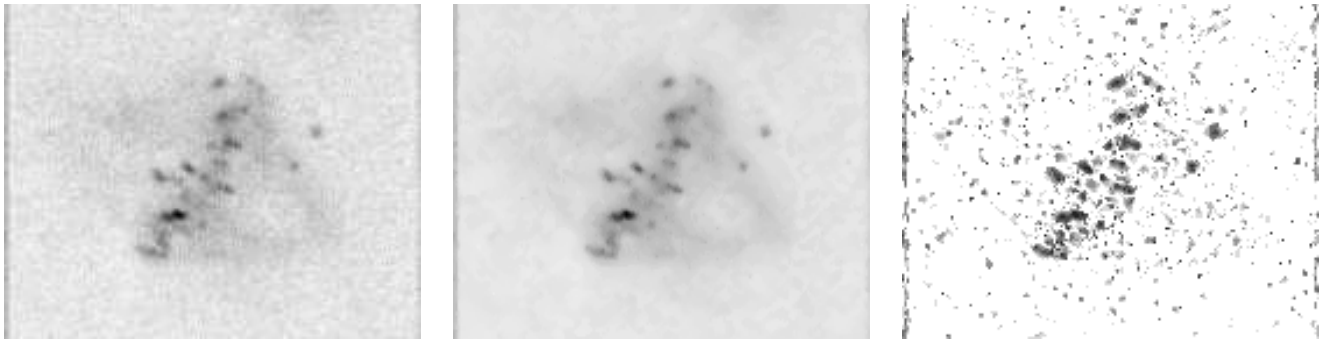
(c) denoised image using Gaussian statistics

Figure 3: Denoising of a confocal image showing an individual neuronal cell.

The performance of the restoration procedure is also demonstrated for a 3D fluorescence microscopic ($179 \times 144 \times 16$) stack. Figure 5a shows a typical 2D image taken from the 3D stack of 16 images depicting moving chromosomes (with dark values) standing over a spatially varying background, during mitosis (courtesy of Institut Curie, Paris, France). Figure 5b shows the image denoised using Poisson statistics, where the location of chromosomes (dark spots) are well preserved. Figure 5c shows also the locations and sizes of optimal estimation windows coded as previously; small windows are mainly in the neighborhood of chromosomes.

5 Conclusion

We have described a novel feature-preserving adaptive algorithm that reduces Poisson noise with a controllable computational complexity. The proposed scheme can be seen also as an alternative method to the anisotropic diffusion and bilateral filtering or energy minimization methods. An advantage of the method is that no hidden parameters need to



(a) original image (b) denoised image (c) locations/sizes of windows

Figure 5: Denoising of a fluorescence microscopic image showing human chromosomes.

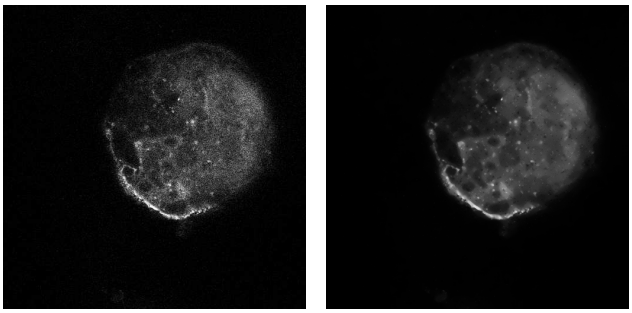


Figure 4: Denoising of a 3D confocal stack depicting membranes of cultured human cells.

be precisely adjusted as in many other methods and the algorithm can be parallelized. Experimental results show its effectiveness and demonstrate its potential for 2D and 3D confocal microscopy image denoising.

References

[1] D. Barash. A fundamental relationship between bilateral filtering, adaptive smoothing and the nonlinear diffusion equation, *IEEE Trans. Patt. Anal. Mach. Intell.*, 24(6): 844-847, 2002.

[2] A. Ben Hamza, H. Krim. A variational approach to maximum a posteriori estimation for image denoising, In *Proc. EMMCVPR'01*, LNCS 2134, pp. 19-34, Sophia-Antipolis, France, 2001.

[3] M.J. Black, G. Sapiro, D. Marimont, D. Heeger. Robust anisotropic diffusion, *IEEE Trans. Image Process.*, 7(3): 421-432, 1998.

[4] D. Comaniciu, P. Meer. Mean-shift: a robust approach toward feature space analysis, *IEEE Trans. Patt. Anal. Mach. Intel.*, 24(5): 603-619, 2002.

[5] I. Csiszár. Why least squares and maximum entropy ? An axiomatic approach to inference for linear inverse problems. *Ann. Statist.*, 19: 2032-2066, 1991.

[6] T. Gasser, L. Sroka, C. Jennen Steinmetz. Residual variance and residual pattern in nonlinear regression, *Biometrika*, 73: 625-633, 1986.

[7] F. Godtliebsen, E. Spjotvoll, J.S. Marron. A nonlinear gaussian filter applied to images with discontinuities, *J. Nonparametric Statistics*, 8: 21-43, 1997.

[8] A. Juditsky, Wavelet estimators: adapting to unknown smoothness, *Math. Methods of Statistics*, No. 1, pp. 1-20, 1997.

[9] O. Lepski. Asymptotically minimax adaptive estimation 1: uppers bounds, *SIAM J. Theory of Prob. and Appl.*, 36(4): 654-659, 1991.

[10] M. Maurizot, P. Bouthemy, B. Delyon, A. Iouditski, J.-M. Odobez. Determination of singular points in 2D deformable flow fields. In *IEEE Int. Conf. Image Processing*, Washington DC, 1995.

[11] R.D. Nowak, R.G. Baraniuk. Wavelet-domain filtering for photon imaging systems, *IEEE Trans. Image Processing*, :666-678, 1999.

[12] J.B. Pawley. Fundamental limits in confocal microscopy, *Handbook of Biological Confocal Microscopy* (ed. by J.B. Pawley), pp. 15-26, Plenum Press, New-York, 1990.

[13] P. Perona, J. Malik. Scale space and edge detection using anisotropic diffusion, *IEEE Trans. Patt. Anal. Mach. Intell.*, 12(7): 629-239, 1990.

[14] J. Polzehl, V. Spokoiny. Adaptive weights smoothing with application to image restoration, *J. Roy. Stat. Soc. B.*, 62(2): 335-354, 2000.

[15] D.L. Snyder, M.I. Miller. *Random Point Process in Time and Space*, Springer Verlag, Berlin, 1991.

[16] J.L. Starck, F. Murtagh, A. Bijaoui. *Image Processing and Data Analysis, the Multiscale Approach* , Cambridge University Press, 2000.

[17] C. Tomasi, R. Manduchi. Bilateral filtering for gray and color images, In *Proc. Int Conf. Comp. Vis. (ICCV'98)*, pp. 839-846, Bombay, India, 1998.

[18] G.M.P. van Kempen, L.J. Van Vliet, P.J. Verveer, H.T.M. van der Voort. A quantitative comparison of image restoration methods for confocal microscopy , *J. Microscopy*, 185(3): 354-365, 1997.

[19] G.M.P. van Kempen, H.T.M. van der Voort, J.G.J Bauman, K.C. Strasters. Comparing maximum likelihood estimation and constrained Tikhonov-Miller restoration, *IEEE. Eng. Med. Biol.*, 15: 76-83, 1996.

Motion Segmentation by quasi-simultaneous multi-parametric Estimation

R. Montoliu and F. Pla

Dept. Lenguajes y Sistemas Informáticos
Jaume I Univerisity
Campus Riu Sec s/n 12071 Castellón, Spain
email:[montoliu,pla]@uji.es, http://www.vision.uji.es

Abstract

This paper presents a framework for the motion segmentation and estimation task on sequences of two grey images without a priori information of the number of moving regions present in the sequence. The proposed algorithm combines temporal information, by using an accurate Generalized Least-Squares Motion Estimation process and spatial information by using an inlier/outlier classification process which classifies regions of pixels, in a first step, and the pixels directly, in a second step, into the different motion models present in the sequence. The performance of the algorithm has been tested on synthetic and real images with multiple objects undergoing different types of motion.

1. Introduction

Segmentation of moving objects in a video sequence is basic task for several applications of computer vision, e.g. a video monitoring system, intelligent-highway system, tracking, airport safety, surveillance tasks and so on. In this paper, Motion Segmentation, also called spatial-temporal segmentation, refers to labelling pixels which are associated with different coherently moving objects or regions in a sequence of two images. Motion Estimation refers to assigning a motion vector to each region (or pixel) in an image.

Although the Motion Segmentation and Estimation problem can be formulated in many different ways ([**?**], [**?**], [**?**], [**?**]), we choose to approach this problem as a multi-structural parametric fitting problem. In this context, the segmentation problem is similar to robust statistical regression. The main difference is that robust statistical regression usually involves statistics for data having one target distribution and corrupted with random outliers. Motion segmentation problems usually have more than one population with distinct distributions and not necessarily with a population having absolute majority.

The problem of fitting an a priori known model to a set of noisy data (with random outliers) was studied in the statistical community for a number of decades. One important contribution was the Least Median of Squares (LMedS) robust estimator but it has the break down point of 50%. This means that LMedS technique needs the population recovered to have at least a majority of 50% (plus 1). Other robust estimators have been developed in order to overcome this problem, which is frequently encountered in different computer vision tasks. They are Adaptive Least k-th Order residual (ALKS) [**?**] and Minimum Unbiased Scale Estimator (MUSE) [**?**]. These techniques minimize the k-th order statistic of the square residuals where the optimum value for the k is determined from the data. The problem of both techniques is the estimation of the correct value of k suffers high computation effort. Bab-Hadiashar and Suter presented a method named Selective Statistical Estimator (SSE) [**?**] which is a variation of the Least K-th order statistic data regression where the user proposes the value k as the lower limit of the size populations one is interested in. All the Motion Segmentation LKS-based algorithms start selecting an initial model using random sampling, and classifying all the pixels into this model using a scale measure. With the remaining pixels the process is repeated until all the pixel have been classified. The main problem of these algorithms is that there are frequently pixels that can be more suitable to belong to a model but they have been classified in an earlier model.

Danuser and Stricker [**?**] presented a similar framework for parametric model fitting. Their algorithm has a fitting step that is one component of the algorithm which also collect model inliers, detects data outliers and determines the a priori unknown total number of meaningful models in the data. They apply a quasi simultaneous application of a general Least Squares fitting while classifying observations in the different parametric data models. They applied their algorithm to multiple lines and planes fitting tasks. The most important advantages with respect to LKS-based algorithms are the use of an exchange step, that permits change of observations among models, and the use of a inliers/outliers classification process, which increases the accuracy of the

segmentation.

In [**?**] a quasi-simultaneous motion segmentation and estimation method based on a parametric model fitting algorithm was presented. The method accurately estimates the affine motion parameters using a generalized least squares fitting process. It also classifies the pixels into the motion models present in two consecutive frames. This algorithm uses each pixel of the image as observation. It suffers from problems of isolated points because it does not use neighbourhood information and need given good initial models to obtain the final motion segmentation. Nevertheless, it indicates that the quasi-simultaneous application of the inliers/outliers classification algorithm and the accurate motion estimator can be useful to be applied in Motion Segmentation tasks.

This paper presents a Motion Segmentation and Estimation algorithm that, in a first step uses regions of pixels as observations in order to obtain good initial models that in a second step will be improved using each pixel as observation. The use of regions in the first step makes the segmentation more spatial consistent. In addition, the algorithm uses neighbourhood constraints to collect new inliers to the model, only regions (or pixels) that are neighbour of the model are considered to be inliers. This algorithm overcomes the need of a good enough previous segmentation of the models (they are obtained in the first step) and allows extracting the models without a priori information of the number of moving regions present in the sequence.

The rest of the paper is organized as follows: Section 2 explains the complete Motion Segmentation and Estimation algorithm. Section 3 presents a set of experiments in order to verify the results obtained with our approach. Finally, some conclusions drawn from this work are described.

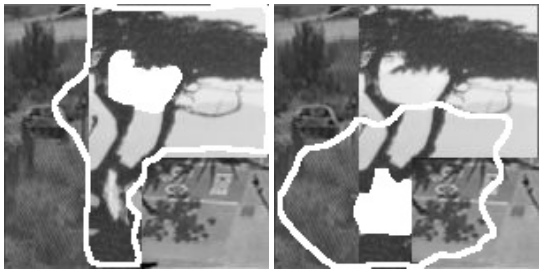
2 Algorithm Outline

In this paper we use the term **Model** as a structure with two elements, the first is a parametric motion vector and the second is a list of regions of the image that support the parametric motion vector. We refer as **Region** to a set of pixels with grey-level coherence.

The input of the algorithm are two consecutive images of a sequence, the first one I_1 captured at time t and the second one I_2 captured at time $t+1$. The output of the algorithm are a motion-based segmentated image I_s and a list of motion parameters corresponding at each region in I_s. For the sake of clarity, we describe the first part of the proposed algorithm in 6 steps:

1. **Preliminaries**: In this step, I_2 is segmented using a given grey level segmentation algorithm. The regions obtained are used as input of the algorithm. An adjacency graph of the previous segmentation is created. In addition the spatial derivates of the images I_1 and I_2 are estimated.

   The purpose of the grey-level segmentation process is to classify the pixels into regions. Our Motion Segmentation algorithm requires that each segmented region should not have pixels belonging to more than one final motion models. Any grey level segmentation algorithm that wherever possible tries to fulfil the previous constraint can be used.



(a) (b)

Figure 1: Two examples of initial models

2. **Get Initial Model**: The aim of this process is find the best possible start point to the global Motion Segmentation and Estimation algorithm. A good initial model is make up of a set of regions that have a high likelihood to belong to the same model. The process starts selecting a region randomly. A model with this region and its neighbours is formed. The motion is estimated for this model using the process in subsection 2.1. A goodness measure GM is calculated for this model. The previous step is repeated q times. The model with the best goodness measure is selected as the initial model.

   The goodness measure is calculated using the following expression: $GM = ((1-l_{avg})*2+(l_{best}-l_{worst}))$ where l_{avg} is the average of the likelihood $L_{M_n}(R)$ for each region R using the motion model M_n (see point 3), l_{best} is the highest likelihood of the regions and l_{worst} is the lowest likelihood of the regions. Therefore, the best initial model is the one which has the less GM.

   Figure 1 shows an illustrative example of two possible initial models for a sequence with three different motion models: static (left part of the image) and two translational motion (the part of the image showing a tree and the bottom right part). The pixels belonging to the region that have been selected randomly have been painted using white color. The limits of the

model made up with the previous region and its possible neighbours are drawn with a continuous white line. Note that in the left image the majority of the pixels perform the same motion (the model of the three) and only a small area performs a different motion. Therefore, its GM will have a very small value. In addition, its GM will be lower than in the case of the right image where there is not a majority of pixels performing the same motion.

3. **Improve the model**: An iterative classification process is started in order to find the inliers and to reject outliers between the k regions that make up the initial model. With the set of resulting regions, we start another classification process with the neighbours of the last inserted regions not yet processed. This process continues until there are not more new neighbour regions to be processed.

   The loop of the inliers/outliers classification consists of:

   (a) Estimate the motion parameters using all the pixels belonging the regions of the model (see subsection 2.1).

   (b) Look for outliers into the regions of the model, if there are outliers, improve the motion parameters. A region R is considered outlier (with respect to model M_n) if the likelihood of region R belonging to a model M_n is lower than a threshold.

   (c) Test each outlier if it can be now considered inlier according the new estimated parameters. If there are new inliers, the parameters are improved again. A region R is considered inlier (with respect to model M_n) if the likelihood of the region R belonging to a model M_n is higher than a threshold.

   (d) Go to step b and repeat until there are not changes in the set of regions of the model.

   In order to estimate a likelihood of a region R belonging to a model M_n, the next expressions are used:

$$L_{M_n}(R) = (\sum_{p_i \in R} L_{M_n}(p_i))/N_R$$
$$L_{M_n}(p_i) = e^{-0.5 * \frac{F^2_{M_n}(p_i)}{\sigma_2}} \quad (1)$$

   where N_R is the number of pixels of the region R. For each pixel p_i belonging to the region R the likelihood $L_{M_n}(p_i)$ of the pixel belonging to a model M_n is calculated. This likelihood ([**?**]) has been modelled as a Gaussian like function where $F_{M_n}(p_i)$ is the residual for the pixel p_i of the objective function using the motion parametric vector of the model M_m.

4. **Exchange of regions**: If a valid model M_n has been extracted, then a region exchange procedure is started. The goal of this procedure is to reclassify regions that have been captured by an early model M_m where $m < n$. A region is moved if it lies closer to the new extracted model and there is a neighbour relationship between the region and the new model. If all the regions of the model M_m lie closer to the new Model M_n then the model M_m is deleted. When for each region of model M_m we can not decide if it lies closer to the model M_m or to the model M_n, then the models are merged, that is, it is considered both models have similar motion parameters.

5. **Repeat**: Go to step 2 and repeat the same process with another initial model if any. If there is any problem estimating the motion of some model, e.g. not enough texture information, not enough number of observations, etc., the regions of this model are moved to a set called *regions with problems* (RWP).

6. **End**: When all possible models have been extracted, the models that only have one region are tested in order to try to merge them with their neighbour models. In addition, each region in the RWP set is tested in order to move it into some of the models in its neighbourhood.

At the end of the first part of the algorithm, a set of NM motion models have been extracted. Each motion model is made up of a vector of parametric motion models and a set of regions which support the motion. Our Motion Segmentation algorithm requires that each region from the given grey-level segmentation should not have pixels belonging to more than one final motion model. It is very likely that some regions will not fulfill this constraint. The second part of the algorithm is performed in order to improve motion segmentation in these regions. In this step, instead of using a region of pixels as observation, each pixel is considered as observation. This process consists of:

1. **Find Outliers**: For each extracted model M_n ($n = 1 \ldots NM$), find all the pixels that can be considered as outliers. They are the pixels p_i which their likelihood respect to the model M_n, $L_{M_n}(p_i)$ is less than a threshold. All the outlier pixels are included in a set, together with the pixels belonging to the region which have been considered outliers in the previous part.

2. **Improve parameters**: The motion parameters for the motion models that have new outliers are improved (see subsection 2.1).

3. **Find Inliers**: For each outlier, test if it can be included in some of the motion models. A pixel p_i will be included in the model with the greatest likelihood $L_{M_n}(p_i)$, $n = 1 \dots NM$, if it is bigger than a threshold and there is a neighbourhood relationship between the pixel p_i and the model M_n.

4. **Improve parameters**: The motion parameters for the motion models that have new inliers are improved (see subsection 2.1).

5. **Repeat**: Repeat 1 to 4 while there are changes in the set of pixels.

At the end of the two parts of the algorithm the pixels have been classified into the different motion models corresponding to the moving objects in the scene. The pixels that could not be included in any model will be considered as outliers.

2.1 Motion Estimation

The Generalized Least Squares (GLS) algorithm is used in order to obtain the motion parameters of a model. The GLS algorithm [**?**] is based on minimizing an objective function O over a set S of r observation vectors, $S = \{L_1, \dots, L_r\}$.

$$O = \sum_{L_i \in S} (F_i(\chi, L_i))^2 \tag{2}$$

where $\chi = (\chi^1, \dots, \chi^p)$ is a vector of p motion parameters and L_i is a vector of n observation $L_i = (L_i^1, \dots, L_i^n)$, $i = 1 \dots r$.

The equation (2) is non-linear, but it can be linearized using the Taylor expansion and neglecting higher order terms. This implies that an iterative solution has to be found. At each iteration, the algorithm estimates $\Delta\chi$, that improves the parameters as follows: $\chi_{t+1} = \chi_t + \Delta\chi$. The increment $\Delta\chi$ is calculated (see [**?**]) using the following expressions:

$$\begin{aligned}
\Delta X &= \left(A^T(BB^T)^{-1}A\right)^{-1} A^T(BB^T)^{-1}W \\
B &= \begin{pmatrix} B_1 & 0 & 0 & 0 \\ 0 & B_2 & 0 & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & B_r \end{pmatrix}_{(r\times(r\times n))} \\
A &= \begin{pmatrix} A_1 \\ A_2 \\ \dots \\ A_r \end{pmatrix}_{(r\times p)} \qquad W = \begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_r \end{pmatrix}_{(r\times 1)} \\
B_i &= \left(\frac{\partial F_i(\chi_t, L_i)}{\partial L_i^1}, \frac{\partial F_i(\chi_t, L_i)}{\partial L_i^2}, \dots, \frac{\partial F_i(\chi_t, L_i)}{\partial L_i^n}\right)_{(1\times n)} \\
A_i &= \left(\frac{\partial F_i(\chi_t, L_i)}{\partial \chi^1}, \frac{\partial F_i(\chi_t, L_i)}{\partial \chi^2}, \dots, \frac{\partial F_i(\chi_t, L_i)}{\partial \chi^p}\right)_{(1\times p)} \\
w_i &= -F_i(\chi_t, L_i)
\end{aligned} \tag{3}$$

In motion estimation problems ([**?**]) the objective function is based on the assumption that the grey level of all the pixels of a region remains constant between two consecutive images. The motion parameters vector, χ, depends on the motion model being used. For each point i, the vector of observation L_i has three elements: column, row and grey level of second image at these coordinates. The objective function is expressed as follows:

$$O = \sum_{L_i \in S} (F_i(\chi, L_i))^2 = \sum_{L_i \in S} (I_1(x_i', y_i') - I_2(x_i, y_i))^2 \tag{4}$$

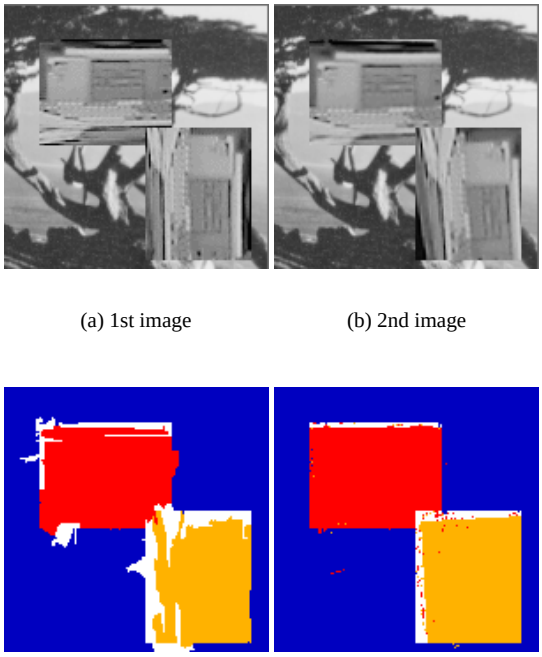
where $I_1(x_i', y_i')$ is the grey level of the first image in the sequence at the transformed point x_i', y_i', and $I_2(x_i, y_i)$ are the grey level of the second image in the sequence at point x_i, y_i. Here, $L_i = (x_i, y_i, I_2(x_i, y_i))$.

The affine motion model is used in this work, which is able to cope with translations, scaling, rotation and shear of images and is defined with a vector of $\chi = (a_1, b_1, c_1, a_2, b_2, c_2)$.

3 Experimental Results

In order to show the performance of the approach presented, two types of experiments have been carry out. In the first experiment, synthetic sequences have been used, where the final motion segmentation and the motion parameters of each model are known. In the second experiment real scenes are used, where the final motion segmentation and the motion parameters are unknown.

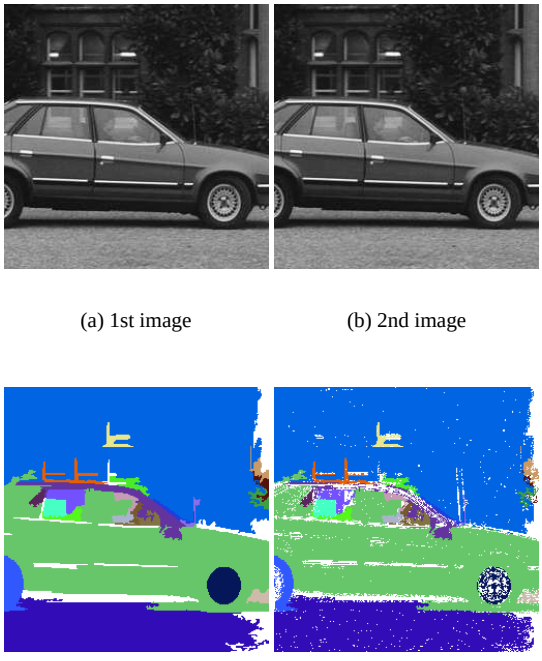
Figures 2(a,b) show both images of an example of synthetic sequence. In this synthetic sequence three different



(a) 1st image (b) 2nd image

(c) Initial gray segmentation (d) Final segmentation

Figure 2: Both images of the synthetic sequence and results



(a) 1st image (b) 2nd image

(c) Initial gray segmentation (d) Final segmentation

Figure 3: Both images of the real sequence and results

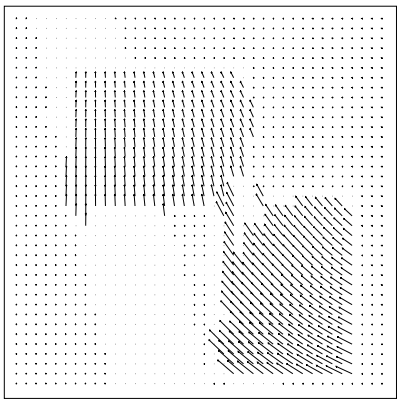


Figure 4: Optic Flow computed from results of the synthetic sequence

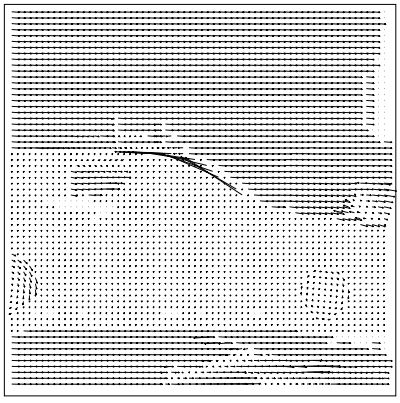


Figure 5: Optic Flow computed from results of the real sequence

motion models can be found. The first one is the background, which performs a null-motion. The second motion model performs a change of scale and the third one corresponds to a rotational motion. Figure 3(a,b) show both images of an example of real sequence.

Figures 2(c,d) and 3(c,d) show the result after the first step of the algorithm and the final results for both sequences. The white pixels in figures 2c and 3c, are the ones that have not been classified in any model. These regions correspond mainly to regions belonging to occluded areas due to the motion and to regions that do not fulfill the requirement of belonging only to a model, i.e. some pixels belong to a model and some other belong to a different model.

Figures 2d and 3d show the segmentation performed after the second step showing how segmentation has been improved in previous regions. Now, white pixels are the ones considered as outliers. They are mainly pixels belonging to occluded areas due to the motion and pixels where our algorithm could not estimate the motion due to lack of texture or to the presence of too large motions.

Figures 4 and 5 show the optic flow for both sequences. They have been computed using the motion parameters of each model in all the pixel belonging to them. They are presented in order to illustrate the motion models estimated.

4 Conclusions

In this paper, a motion segmentation and estimation algorithm has been presented, which can extract different moving regions present in the scene quasi-simultaneously and without a priori information of the number of moving objects. The main properties of our approach are:

- A GLS Motion Estimation is used, which produces accurate estimation of the motion parameters.
- A classification process which collects inliers, rejects outliers and exchanges regions among models allows to improve motion segmentation.
- It uses, in the first step, regions of pixels and neighbourhood information, that improves the spatial consistency and provides a good initial point to start the second step of the algorithm, which using pixels as observations improves the segmentation in the regions.
- The pixels considered as outliers are mainly pixels belonging to occluded areas due to the motion, thus, detection of outliers provides valuable information about occluded areas.

Future work must study hierarchical techniques in order to improve the speed of the algorithm and to cope with larger motion. The possibility of using sequences with more than two images will be also studied.

A Variational Approach to 3D Geometry Reconstruction from Two or Multiple Views

L. Alvarez, C. Cuenca, A. Salgado, J. Sanchez
Dpto. Informatica y Sistemas
Universidad de Las Palmas de Gran Canaria
{lalvarez,ccuenca,a2652,jsanchez}@dis.ulpgc.es

Abstract

In the last years we have developed some methods for 3D reconstruction. First we began with the problem of reconstructing a 3D scene from a stereoscopic pair of images. We developed some methods based on energy functionals which produce dense disparity maps by preserving discontinuities from image boundaries. Then we passed to the problem of reconstructing a 3D scene from multiple views (more than 2). The method for multiple view reconstruction relies on the method for stereoscopic reconstruction. For every pair of consecutive images we estimate a disparity map and then we apply a robust method that searches for good correspondences through the sequence of images. Recently we have proposed several methods for 3D surface regularization. This is a postprocessing step necessary for smoothing the final surface, which could be afected by noise or mismatch correspondences. These regularization methods are interesting because they use the information from the reconstructing process and not only from the 3D surface. We have tackled all these problems from an energy minimization approach. We investigate the associated Euler–Lagrange equation of the energy functional, and we approach the solution of the underlying partial differential equation (PDE) using a gradient descent method.

1 Introduction

This paper is about three different main topics: The first topic is the 3D reconstruction from two views. The second is the problem of reconstructing a 3D scene from multiple views – more than two – that, in our case, make use of the previous method. The third one deals with the problem of regularizing a 3D surface. This method takes into account the information from the two previous topics.

For the first problem we present a variational approach to recover a dense disparity map from a set of two *weakly calibrated* stereoscopic images. To solve this problem, we first make full use of the knowledge of the so-called *fundamental matrix* to derive the equations that relate corresponding pixels in the two views, and then combine regularization and scale-space tools to estimate iteratively and hierarchically the disparity map. The solution obtained at a coarse spatial scale is used to restrict searching at finer scales. We minimize an energy term that takes into account the epipolar line constraint as well as the edge information constraint through an appropriate regularization term. In order to reduce the risk to be trapped within some irrelevant local minima during the iterations, we use a focusing strategy based on a linear scale-space. This method is explained in paper [1]. We have also implemented a symmetric method for computing the disparity map in both senses (see [4]). In papers [2] and [3] we have proposed methods for computing the optical flow between two images which are very similar to the disparity map estimation method except that we do not use any geometric constraint as the epipolar geometry.

For the second problem – reconstruction from multiple views – we have developed a robust method to recover a 3D model. After computing the disparity maps for every two consecutive frames we search for the best sequences of corresponding points through the set of frames. We estimate sequences of corresponding points across the multiple view image sequence. Basically, we try to connect points between images following the disparity map estimation. We select sequences of correspondent points for which the forward and backward disparity estimations are coherent. That is, if we take the initial point and go through the sequence using the forward disparity estimations and then go back using the backward disparity estimations we have to arrive to the same point (modulus a threshold parameter). From each selected corresponding point sequence we recover a 3D point by intersecting the projection lines of the points in the sequence. By collecting the 3D points obtained from each sequence we recover an unstructured

set of 3D points. Recently, a new accurate technique based on a variational approach has been proposed in [17, 18]. Using a level set approach, this technique optimizes a 3D surface by minimizing an energy that takes into account the surface regularity as well as the projection of the surface on different images.In this paper we propose a different approach which is also based on a variational formulation but only using a disparity estimation between images and without defining explicitly any 3D surface.

For the third problem we present a method for the regularization of a set of unstructured 3D points obtained from a sequence of stereo images. Typically, the recovered set of 3D points is noisy, because of errors in the camera calibration process, errors in the disparity estimations, errors in the corresponding point sequences computations, etc., so some kind of regularization is needed. The regularization model we propose is a variational approach. We propose a model based on an energy. This method takes into account the information supplied by the disparity maps computed between pair of images to constraint the regularization of the set of 3D points. As in the first problem there is a regularization term that relies on an operator that is very similar to the Nagel–Enkelmann operator which allows for the regularization of the set of 3D points by preserving discontinuities presented on the disparity maps. One interesting advantage of this approach is that we regularize the set of 3D points by only using the 2D image projection information and, in particular, we do not need to define any 3D triangulation on the set of 3D points. In paper [5] we proposed a general method for regularizing a set of 3D points according to the information of the disparity maps, and in paper [6] there is an explanation of a 3D regularization method for cylindrical surfaces. In this paper we present the first approach for general surfaces.

The paper is organized as follows: In Section 2.1, we present the model for stereoscopic reconstruction from two views. In Section 2.2 we explain the 3D reconstruction method from multiple views. In Section 3 we present the regularization model and in Section 4 the conclusions.

2 Reconstruction model

2.1 Reconstruction from two views

In order to estimate a dense disparity map between two images we present an energy based approach. This energy also preserves discontinuities resulting from image boundaries. We derive a simplified expression for the disparity that allows us to easily estimate it from a stereo pair of images using an energy minimization approach. We assume that the epipolar geometry is known, and we include this information in the energy model. Discontinuities are preserved by means of a regularization term based on the Nagel–Enkelmann operator. We investigate the associated Euler–Lagrange equation of the energy functional, and we approach the solution of the underlying partial differential equation (PDE) using a gradient descent method. In order to reduce the risk to be trapped within some irrelevant local minima during the iterations, we use a focusing strategy based on a linear scale-space.

In order to estimate the disparity ($\lambda(x,y)$), one can proceed in a classical way and try to recover this important information using a simple correlation scheme. Unfortunately, this naive solution will not provide a correct and accurate solution, in particular in the regions where the disparity map may present some discontinuities, as is often the case close to image edges. It is well known that the disparity map obtained using this classical method tends to be very smooth across the boundaries of the images. The idea we would like to formalize and develop here is to estimate a $\lambda(x,y)$ function which is smooth only along the image boundaries and not across them. This leads us to consider the minimization of the following energy functional:

$$\begin{aligned} E(\lambda) &= \int_\Omega \left(I_l(x,y) - I_r(x+u(\lambda), y+v(\lambda))\right)^2 dxdy \\ &+ C\int_\Omega \Phi\left(\nabla I_l, \nabla\lambda\right) dx\, dy \end{aligned} \quad (1)$$

where Ω is the image domain, C is a positive constant, and $\Phi(\nabla I_l, \nabla\lambda)$ determines the regularization term. This function includes a diffusion tensor first proposed by Nagel and Enkelmann that guides the diffusion along the contours at image boundaries and in all directions at homogeneus regions. The associated Euler-Lagrange equations give us a diffusion PDE, which is then embedded into a gradient descend process to reach the solution:

$$\begin{aligned} \frac{\partial\lambda}{\partial t} &= C\,\mathrm{div}\left(D\left(\nabla I_l\right)\,\nabla\lambda\right) + \left(I_l(x,y) - I_r^\lambda(x,y)\right) \\ &\quad \frac{a\left(\frac{\partial I_r}{\partial y}\right)^\lambda(x,y) - b\left(\frac{\partial I_r}{\partial x}\right)^\lambda(x,y)}{\sqrt{a^2+b^2}}. \end{aligned} \quad (2)$$

In Figure 1 we show an example of disparity map computation from a stereoscopic pair of a human face. We also show the result given by a common correlation based technique.

In Figure 2 we show four views of the 3D reconstruction of the previous stereo pair.

Figure 1: Top: the original stereo pair. Bottom left: the computed disparity map using a correlation window of size 13×13. Bottom right: the result from our method, with the correlation result as initialization.

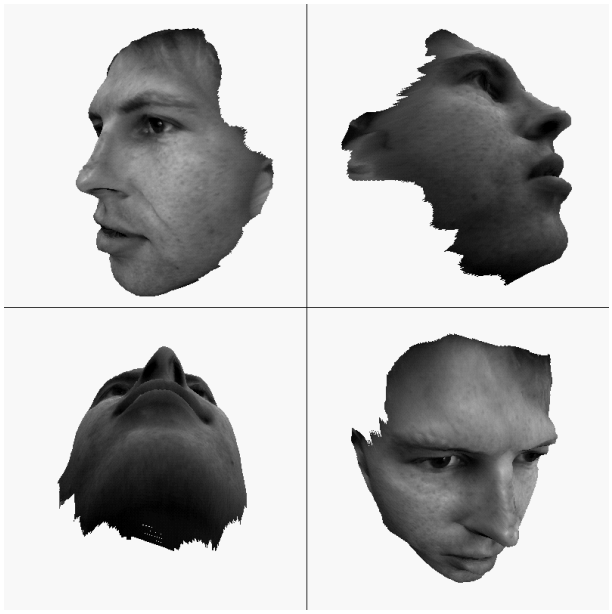


Figure 2: Four views of the 3-D reconstruction of the stereo pair in Figure 1, using the disparity map from our method.

2.2 Reconstruction from several views

We have developed a robust method to recover a 3D model from several views. All the views have been taken at the same time and all of them pointing to a common 3D scene. We make use of the previous model for stereoscopic images. This method follows these steps:

- For each pair of consecutive images, we estimate a dense disparity map using the accurate technique developed in [1]. We estimate such disparity map forward and backward, that is, from one image to the next one and in the opposite direction.
- We estimate sequences of corresponding points across the multiple view image sequence. Basically, we try to connect points between images following the disparity map estimation. We select sequences of correspondent points for which the forward and backward disparity estimations are coherent. That is, if we take the initial point and go through the sequence using the forward disparity estimations and then go back using the backward disparity estimations we have to arrive to the same point (modulus a threshold parameter). We keep trace of the pixels belonging to a sequence in order to avoid that the same pixel is included in different groups.
- From each selected corresponding points sequence we recover a 3D point by intersecting the projection lines of the points in the sequence. By collecting the 3D points obtained from each sequence we recover an unstructured set of 3D points.
- Typically, the recovered set of 3D points is noisy, because of errors in the camera calibration process, errors in the disparity estimations, errors in the corresponding point sequences computations, etc., so some kind of regularization is needed. In this paper, we propose a new variational model to smooth the unstructured set of 3D points. This regularization model is based on the 2D image information and does not require to define any kind of geometric relation between the 3D points.

In Figure 4, we show the front and profile views of the reconstruction of the Bust sequence.

3 Regularization model

In this section we present a method for the regularization of a set of unstructured 3D points obtained from a sequence of stereo images. This is a postprocessing step to the reconstruction from multiple views explained in the previous section. This method takes into account the information supplied by the disparity maps computed between pair of images to constraint the regu-

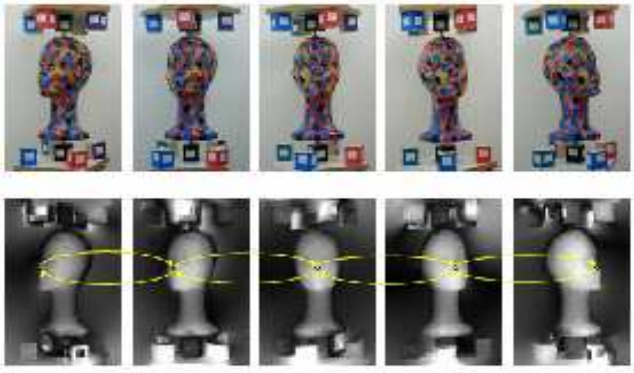


Figure 3: Some images of the Bust sequence and the corresponding disparity maps associated to them. We search for sequences of corresponding points which have a small error going forward and backward, therefore we have to compute disparity maps for every pair of images in both senses.

Figure 4: Front and profile views of the 3D reconstruction of the Bust sequence.

larization of the set of 3D points. We propose a model based on an energy which is composed of an attachment term that minimizes the distance from 3D points to the projective lines of camera points, and a second term which relies on an operator that is very similar to the Nagel–Enkelmann operator. This second term allows for the regularization of the set of 3D points by preserving discontinuities presented on the disparity maps. One interesting advantage of this approach is that we regularize the set of 3D points by only using the 2D image projection information and, in particular, we do not need to define any 3D triangulation on the set of 3D points. We embed this energy in a 2D finite element method to take advantage of the underlying float precision mesh. Delauny triangulations are managed to give support to this finite element method.

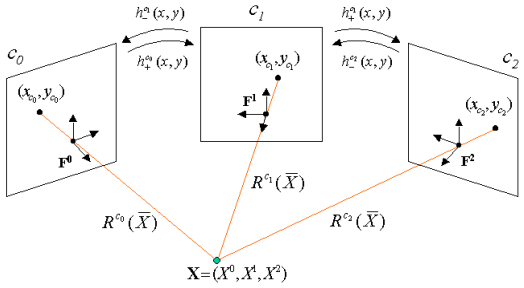


Figure 5: Notation

After minimizing, this 2D finite element method results in a large system of equations that can be optimized for fast computations over no-null values. We derive an efficient implicit numerical scheme which reduces the number of calculations and memory allocations.

The energy to be minimized for the regularization of the set of 3D point is:

$$E\left(\bar{X}^0.,\bar{X}^{N_c-1}\right) = \sum_{c=0}^{N_c-1}\left(\int_\Omega dist(\bar{X}^c,\bar{R}^c)^2 + \alpha \sum_{i=0}^{2}\int_\Omega \nabla^T X^c D(\bar{h}^c)\nabla X^c\right) \quad (3)$$

where α is a parameter that states the balance between the two terms and $dist(\bar{X}^c,\bar{R}^c)$ denotes the distance from point $\bar{X}^c$ to the straight line $\bar{R}^c$ and is given by formula

$$dist(\bar{X}^c,\bar{R}^c)^2 = \sum_{i=0}^{2}\left(X^{c,i}-F^{c,i}\right)^2 - \left(\sum_{i=0}^{2} l^{c,i}\left(X^{c,i}-F^{c,i}\right)\right)^2 . \quad (4)$$

In Figure 5 is the notation that we have used for equations (3) and (4) and in Figure 6 we show the result of applying this method to the 3D reconstruction on Figure 4.

4 Conclusions

Our method for 3D reconstruction from a pair of stereoscopic images combines some techniques developed in the context of optic flow estimation [2, 22] with some other techniques developed in the context of dense disparity map estimation which take into account the geometric constraints associated to a stereo pair. We think

Figure 6: Front and profile views of a regularization of the 3-D reconstruction on Figure 4

that the combination of these ideas is fruitful in that it produces new tools to estimate dense disparity fields which benefit from the research efforts in stereo vision as well as in optic flow estimation.

The method for 3D reconstruction from multiple views is very robust in the sense that it intensively looks for the best matching sequences of points. This yields a set of 3D points that are probably the most accurate ones and discards those points that have not good matches or do not appear in enough views – maybe due to occlusions.

In this paper we have presented a novel method for the regularization of a set of 3D points. We have established an energy in a traditional attachment–regularizing couple of terms. In the regularizing term we have made used of an operator similar to the Nagel–Enkelmann operator for 3D regularizations.

This energy model has been embedded into a 2D finite element approach to take advantage of the underlying precision of data. Then we have managed to derive this energy and propose a very efficient and optimal numerical scheme that allows us to speed up the process and reduce the memory needs.

One of the main advantages of the method is that it regularizes sets of unstructured 3D points without using any geometric relation in 3D. We only use the information of the projection of the points in the cameras. In particular, this method could be used as a preprocessing step before the construction of 3D surfaces fitting the 3D points. We notice that most of the techniques for such surface reconstruction are very sensitive to the noise in the 3D points representation, and they require the set of 3D points to be regular enough to work properly.

5 Acknowledgments

This work has been partially supported by the research project PI2002/193 founded by the Consejería de Educación, Cultura y Deportes of the Canary Islands Government.

References

[1] L. Alvarez, R. Deriche, J. Sánchez and J. Weickert, *Dense disparity map estimation respecting image discontinuities: a PDE and scale-space based approach*, Journal of Visual Communication and Image Representation, 13:3–21, January 2002. Also published as Inria Research Report n^o 3874

[2] L. Alvarez, J. Weickert, J. Sánchez, *A scale-space approach to nonlocal optical flow calculations*, M. Nielsen, P. Johansen, O.F. Olsen, J. Weickert (Eds.), Scale-space theories in computer vision, Lecture Notes in Computer Science, Springer, Berlin, Vol. 1682, 235–246, 1999.

[3] L. Alvarez, J. Weickert, J. Sánchez, *Reliable Estimation of Optical Flow for Large Displacements*, International Journal of Computer Vision, 39(1):41–56, 2000. An extended version maybe be found at Technical Report No. 2, Cuadernos del Instituto Universitario de Ciencias y Tecnologías Cibernéticas, Universidad de Las Palmas de Gran Canaria, Spain, 1999

[4] L. Alvarez, R. Deriche, T. Papadopoulo and J. Sánchez, *Symmetrical dense optical flow estimation with occlussions detection*, ECCV 2002 Congress (Lectures Notes in Computer Science), Cophenaghe, Denmark, May 2002

[5] L. Alvarez, C. Cuenca, J. Sánchez, *Regularizing a Set of Unstructured 3D Points from a Sequence of Stereo Images*, Scale-Space 2003 Congress, Isle of Skye, England, June 2003

[6] L. Alvarez, C. Cuenca, J. Sánchez, *Regularization of 3D Cylindrical Surfaces*, IBPRIA 2003 Congress, Palma de Mallorca, Spain, June 2003

[7] P. Charbonnier, L. Blanc–Féraud, G. Aubert, M. Barlaud, *Two deterministic half-quadratic regularization algorithms for computed imaging*, Proc. IEEE Int. Conf. Image Processing (ICIP–94, Austin, Nov. 13–16, 1994), Vol. 2, IEEE Computer Society Press, Los Alamitos, 168–172, 1994.

[8] Frédéric Devernay and Olivier D. Faugeras. Computing differential properties of 3-D shapes from stereoscopic images without 3-D models. Proc. IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, June 21–23, 1994, pages 208–213.

[9] W. Enkelmann, *Investigation of multigrid algorithms for the estimation of optical flow fields in image sequences*, Computer Vision, Graphics and Image Processing, Vol. 43, 150–177, 1988.

[10] Olivier Faugeras, Bernard Hotz, Hervé Mathieu, Thierry Viéville, Zhengyou Zhang, Pascal Fua, Eric Théron, Laurent Moll, Gérard Berry, Jean Vuillemin, Patrice Bertin, and Catherine Proy. Real time correlation based stereo: algorithm implementations and applications. *International Journal of Computer Vision*. To appear.

[11] D. Geman, G. Reynolds, *Constrained restoration and the recovery of discontinuities*, IEEE Trans. Pattern Anal. Mach. Intell., 14, 367–383, 1992

[12] D.G. Jones, J. Malik, *A computational framework for determining stereo correspondence from a set of linear spatial filters*, G. Sandini (Ed.), Computer vision – ECCV '92, Lecture Notes in Computer Science, Vol. 588, Springer, Berlin, 395–410, 1992.

[13] N. Amenta and M. Bern. Surface Reconstruction by Voronoi Filtering. *14th ACM Symposium on computational Geometry*, pages 248–253, June 1998

[14] F. Bernardini, J. Mittleman and H. Rushmeier and C. Silva. The Ball Pivoting Algorithm for Surfarce Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, Oct.–Dec. 1999

[15] H. Edelsbrunner and E. P. Mucke. Three-dimensional Alpha Shapes. *ACM Transactions on Graphics*, 13(1):43–72, January 1994

[16] O. Faugeras. Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press, 1993

[17] O. Faugeras and R. Keriven. Complete Dense Stereovision Using Level Set Methods. *Proceedings of Fifth European Conference on Computer Vision*, 1998

[18] O. Faugeras and R. Keriven. Variational principles, surface evolution, PDEs, level set methods and the stereo problem. *IEEE Transactions on Image Processing. Special issue on Geometry driven diffusion and PDEs in image processing*, 7(3):336–344, 1998

[19] O. Faugeras, and Q. Luong and T. Papadopoulo. The Geometry of Multiple Images. Mit Press, 2001

[20] P. George and H. Borouchaki. Triangulation de Delaunay et Maillage. Hermes, 1997

[21] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2000

[22] H.H. Nagel and W. Enkelmann. An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Images Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(5):565–593, 1986

Multiscale Analysis of Natural Textures

Miguel Alemán and Luis Alvarez
Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
Campus de Tafira, 35017, Las Palmas
{maleman,lalvarez}@dis.ulpgc.es

ABSTRACT

Textures are an important feature when we try to identify or classify elements in a scene. Analyzing the textures in a given environment at different scales provides more information than considering the features which can be extracted from a single one. Thus, the relations between the simplest, the most complex or the intermediate patterns which are present in the texture can be studied in the texels, or repeated elements which constitute a textured region.

However, there are some aspects which differ from perfectly generated artificial textures and natural ones. In the first class, the information is limited by the generation of the texture and an increase or decrease in the scale within certain bounds does not produce a change in the elements, but in its resolution. On the other hand, natural textures produce, when the scale is increased, a parallel change in the elements which are visible and thus, in the information which can be extracted, since some elements appear while other disappear.

In this paper, we present a multiscale approach for classifying and indexing textures, and some considerations on the analysis of natural textures, which must be taken into account when analyzing similar scenes at different resolutions, are introduced.

1 Introduction

In this paper, we present a method for texture analysis based on the distribution of the orientation of the edges across textured regions. To this end, we estimate the gradient in every point of the image and build an orientation histogram to describe it. This allows performing satisfactory classifications in most cases, but some of them are not properly classified. A multiscale analysis of the textures improves the results, considering the evolution of the textures along the scale. In natural textures, the changes produced when a certain scene is observed at different distances introduce new elements which must be taken into account when comparing the views. This method and the properties of natural textures have been tested with two different databases of textured images.

The paper in structured as follows: Section 2 shows how textures can be described through their orientation histograms. Section 3 describes the classification of textures from their orientation histograms. In section 4, multiscale analysis is introduced to improve the classification method, and some considerations are analyzed in natural textures. Finally, section 5 presents some conclusions of the work.

2 Texture Description through Orientation Histograms

In order to describe a texture in terms of the orientations of the edges which are present in it, we must estimate the magnitude and the orientation of the gradient in every point of the region. With these values, we can build an orientation histogram which reflects what the relative importance of every orientation is. We have used the structure tensor method to estimate both and build the orientation histogram.

In order to estimate the magnitude of the gradient, we first calculate an initial estimation for every point using the following mask for the horizontal component u_x:

$$\frac{1}{4h}\begin{pmatrix} -(2-\sqrt{2}) & 0 & (2-\sqrt{2}) \\ -2(\sqrt{2}-1) & 0 & 2(\sqrt{2}-1) \\ -(2-\sqrt{2}) & 0 & (2-\sqrt{2}) \end{pmatrix} \quad (1)$$

and the following one for the vertical component u_y:

$$\frac{1}{4h}\begin{pmatrix} -(2-\sqrt{2}) & -2(\sqrt{2}-1) & -(2-\sqrt{2}) \\ 0 & 0 & 0 \\ (2-\sqrt{2}) & 2(\sqrt{2}-1) & (2-\sqrt{2}) \end{pmatrix} \quad (2)$$

If we have an estimation of the gradient (x_n, y_n) for a certain point, the vector (a, b) would be parallel to that gradient if the following product is 0:

$$ay_n - bx_n = (a,b)(x_n,y_n)^{\perp} = \|(a,b)\|\,\|(x_n,y_n)\|\cos(\alpha) \quad (3)$$

where α is the angle formed by the directions of $(x_n, y_n)^{\perp}$ and (a, b). Thus, if we want to find the direction which best represents the estimations for a set of points, we must calculate the vector (a, b) which minimizes the following sum:

$$(a,b)\begin{pmatrix} \sum_{i=0}^{N} y_i^2 & -\sum_{i=0}^{N} x_i y_i \\ -\sum_{i=0}^{N} x_i y_i & \sum_{i=0}^{N} x_i^2 \end{pmatrix}\begin{pmatrix} a \\ b \end{pmatrix} \quad (4)$$

For every component of the sum in the matrix, there are two eigenvalues and two eigenvectors. One of the eigenvalues is 0, and its corresponding eigenvector is the estimation of the orientation of the gradient (x_i, y_i):

$$\begin{pmatrix} y_i^2 & -x_i y_i \\ -x_i y_i & x_i^2 \end{pmatrix}\begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (5)$$

Since any two-dimensional vector can be expressed as a linear combination of the two linearly independent eigenvectors, the maximum variation will be achieved by the eigenvector associated to the maximum eigenvalue. We can see that this vector is $(-y_i, x_i)$, and its corresponding eigenvalue is $\left(x_i^2 + y_i^2\right)$, i.e. the square of the norm of the gradient estimation:

$$\begin{pmatrix} y_i^2 & -x_i y_i \\ -x_i y_i & x_i^2 \end{pmatrix}\begin{pmatrix} -y_i \\ x_i \end{pmatrix} = \left(x_i^2 + y_i^2\right)\begin{pmatrix} -y_i \\ x_i \end{pmatrix} \quad (6)$$

Consequently, $\left(x_i^2 + y_i^2\right)$ is the highest eigenvalue, and its square root is the norm of the gradient. The orientation can be estimated with the eigenvector associated to the lowest eigenvalue of the matrix.

Once the orientation and the magnitude of the gradient have been calculated in every point of the textured region, by adding the magnitude in the points with the same orientation, we can build the orientation histogram for such texture, as shown in Fig. 1.

3 Orientation-Based Texture Classification

In order to relate two textures, an energy function is built, in which the Fourier coefficients of both histograms are compared. We must achieve rotational invariance, in the sense that the result must not be affected if the textures are rotated. A change in the orientation of a texture will only cause a cyclical shift in the histogram. For this reason, the

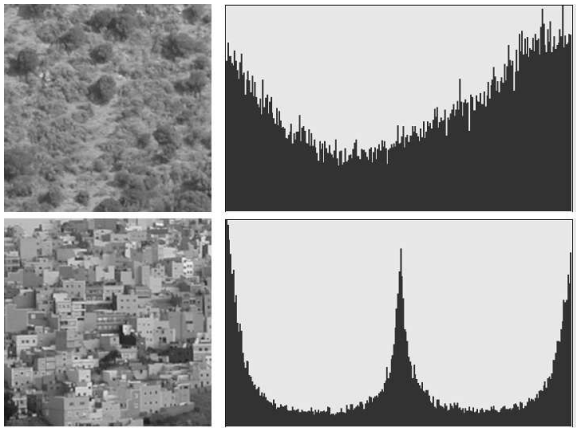


Figure 1: Examples of two textures and their corresponding orientation histograms

Fourier coefficients are modified as follows: let f_n and g_n be the orientation histograms of length L corresponding to the same texture but shifted a positions, i.e. the texture has been rotated an angle $\theta = 2\pi a/L$, and let f_k and g_k be the k^{th} Fourier coefficients of these histograms, then $f_k = g_k e^{-i\frac{2\pi k a}{L}}$. Thus, a measure of how similar the coefficients of both textures are is given by:

$$E(a) = \sum_{k=1}^{\frac{L}{2}} \left(f_k - g_k e^{-i\frac{2\pi k a}{L}}\right)\left(f_k - g_k e^{-i\frac{2\pi k a}{L}}\right)^{*} \quad (7)$$

In addition, the fact that the number of discrete orientations used for the histograms is constant and the normalization of the weights make the lengths of the signals and the total weight equal in both textures. Consequently, a change in the size of the region where the texture is analyzed will not cause the generation of a different distribution. Due to the fact that the higher frequencies are more affected by noise than the lower ones, a monotonic decreasing weighting function $w(.)$ can be used to emphasize the discrimination, thus obtaining the following expression, in which the first terms have a more important contribution than the last ones:

$$E(a) = \sum_{k=1}^{\frac{L}{2}} w\left(\frac{2k}{L}\right)\left(f_k - g_k e^{-i\frac{2\pi k a}{L}}\right)\left(f_k - g_k e^{-i\frac{2\pi k a}{L}}\right)^{*} \quad (8)$$

To test this technique, we have used two sets of textures contained in two databases (see examples in Fig. 2 and Fig. 3). The first database has been made publicly available for research purposes by Columbia and Utrecht Universities, Columbia-Utrecht Reflectance and Texture Database

[2]. We work with grayscale images and thus, a single histogram is used to represent the orientations of the edges in light intensity. If we were working with color images, a three subchannel mechanism could be used when hue, and not only intensity, is relevant for texture identification. Using the techniques previously described, a certain texture is compared with all those in the database and the most similar ones are selected. The similarity between two textures is given by the energy obtained when comparing their orientation histograms.

In Fig. 4 and Fig. 5, and in tables 1 and 2, we show some results of the application of the technique explained above. From the image databases, one is selected and the 5 best comparisons are shown. Of course, as the selected image belongs to the set, the best match corresponds to itself, and the energy factor is 0.

order	txt. number	wtd. energy
1	51	0.00000
2	40	1.65841
3	56	1.99648
4	27	3.74986
5	12	4.10221

Table 1: Results for texture 51 in database 1

order	txt. number	wtd. energy
1	25	0.00000
2	29	0.09970
3	26	0.18930
4	34	0.26078
5	33	0.30463

Table 2: Results for texture 25 in database 2

As mentioned before, the orientation histograms extracted from the textures describe how the different orientations are quantitatively distributed across the region which is studied, but they do not provide any information about the spatial neighborhood of the pixels with a certain orientation. Thus, a completely noisy image, in which all orientations are equally but disorderly present in the image would generate a similar histogram than a circle, where the orientation is gradually increased along its outline. This forces us to search for a certain technique which complements the information provided by this kind of histograms in order to enhance the recognition capabilities.

A multiscale analysis of the images will provide us with a series of images which represent the evolution of each texture at different scales. In this evolution, the orientations will be differently affected by the others, depending on their spatial proximity. This will allow us to distinguish

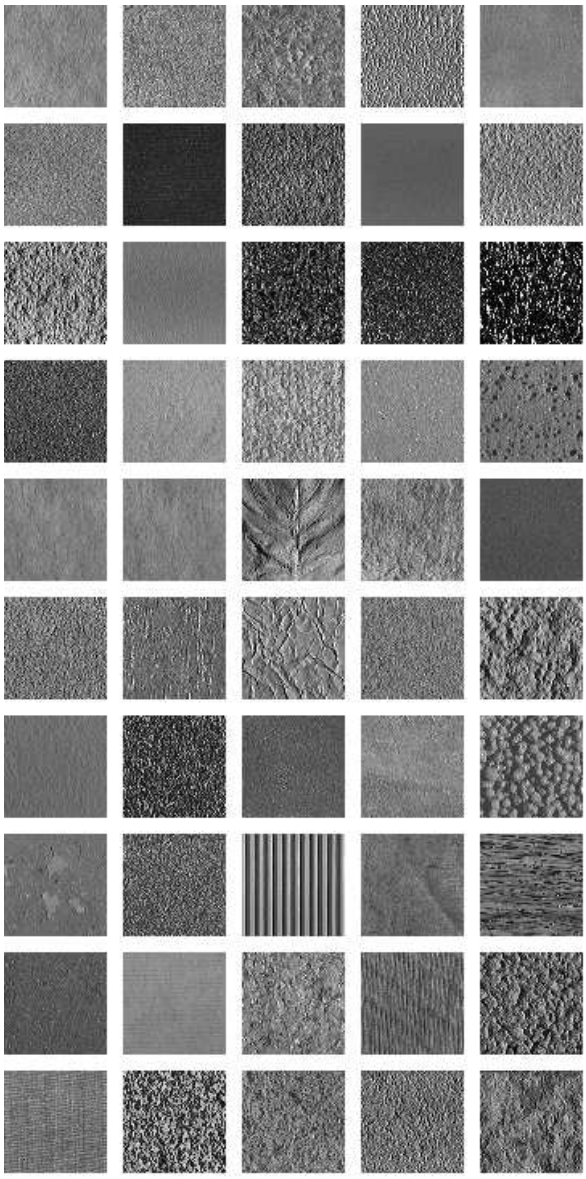


Figure 2: Examples of textures of different materials

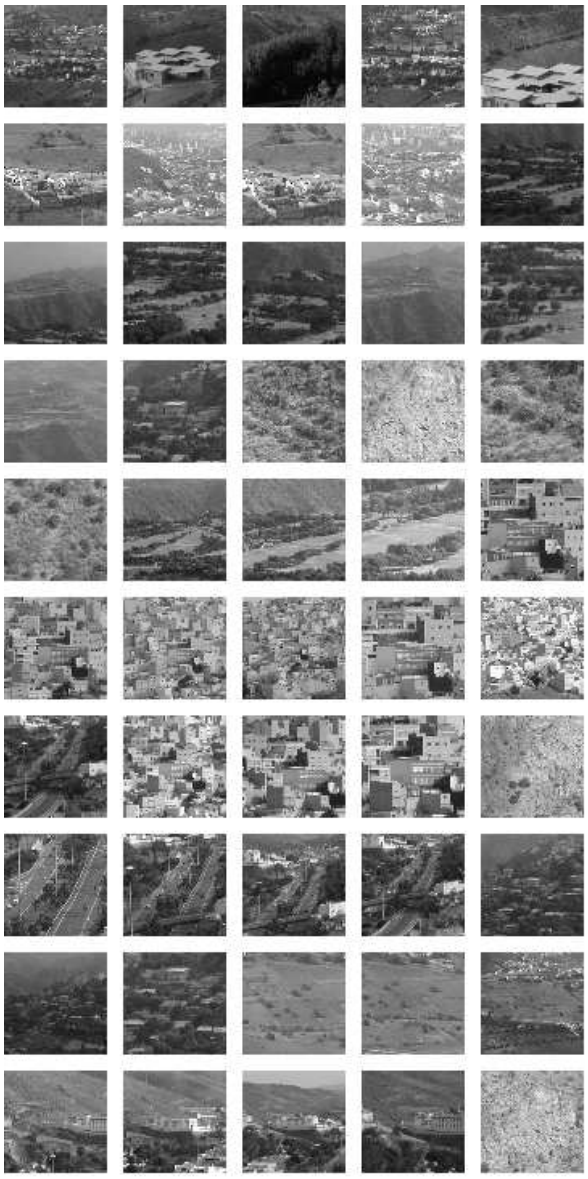


Figure 3: Texture database with natural pictures

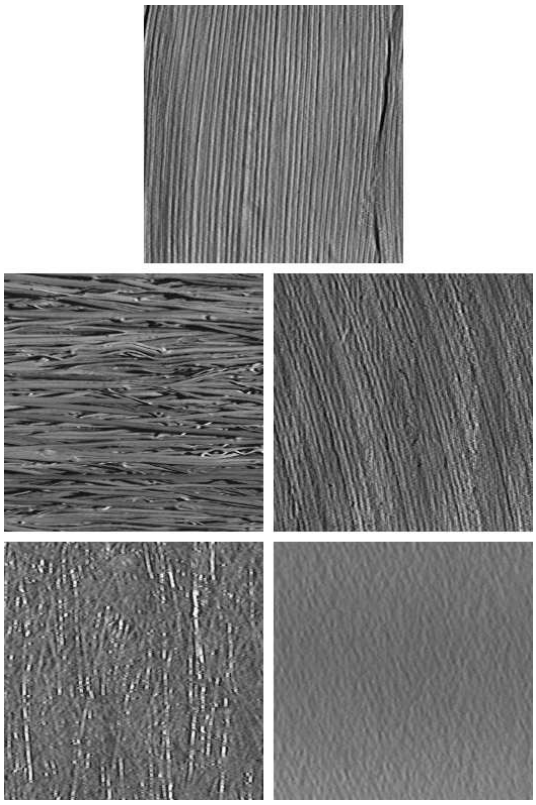


Figure 4: Examples of similar textures extracted for a texture in database 1

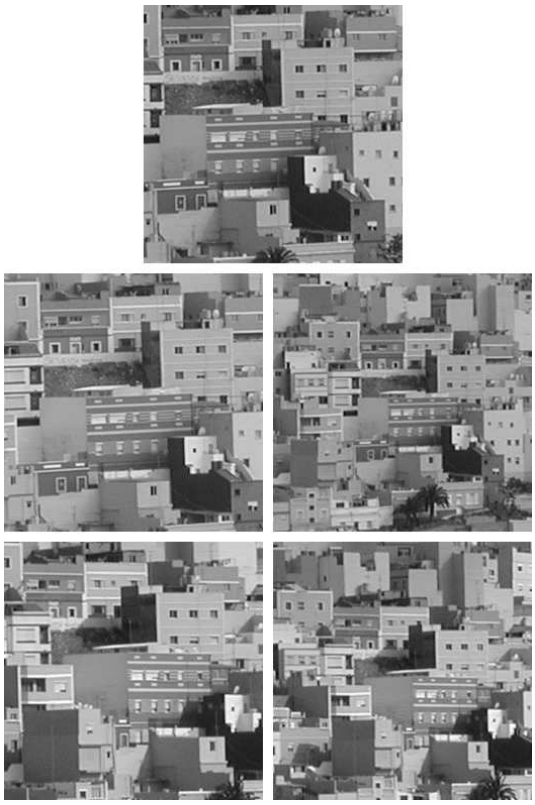


Figure 5: Examples of similar textures extracted for a texture in database 2

among textures where orientations are originally distributed in a similar way, but which are actually different.

4 Multiscale Texture Analysis

The interpretation of the information we perceive from the environment depends on the scale we use to process it. At the same time, the information provided by each scale is useful and the study of the same scene at different scales makes it possible to perceive a wider range of realities. Furthermore, elements which are not distinguishable at a certain scale may be clearly distinct at a different one and the rough and detailed information extracted from an image may help us decide when comparing textures. The multiscale analysis approach has been successfully used in the literature for texture enhancement and segmentation (see [5] and [6] for more details).

A multiscale analysis can be determined by a set of transformations $\{T_t\}_{t\geq 0}$, where t represents the scale. Let I be an image, i.e. $I : \Omega \longrightarrow \Re$, where Ω is the domain where the image is defined. In what follows, we will consider for simplicity in the exposition that $\Omega = \Re^n$ and $I \in H^2(\Omega)$ that is, I and ∇I have finite L^2 norm. $I_t = T_t(I)$ is a new image which corresponds to I at a scale t. For a given image I, to which the multiscale analysis is applied, we can extract a histogram $\{h_i^t\}_{i=0,..,L-1}$ which determines the distribution of the orientations of I at the scale t. In this case, the normalization of the values within a histogram is performed with respect to the initial addition, and not with respect to the addition at that scale. In order to compare the histograms of two images, the scale must be first adjusted.

4.1 Gaussian Multiscale Analysis

As said before, a multiscale analysis generates, for a given image, a series of images which show the evolution of the input signal when a certain process is applied. We will use a Gaussian filter, whose properties are described in [3] and [4]. In one dimension, we use the following Gaussian kernel, where the scale t is related to the standard deviation σ according to the expression $2t = \sigma^2$:

$$K_t(x) = \frac{1}{\sqrt{4\pi t}} e^{-\frac{x^2}{4t}} \tag{9}$$

$$T_t(f)(x) = \int_{\Re} \frac{1}{\sqrt{4\pi t}} e^{-\frac{(x-y)^2}{4t}} f(y)dy \tag{10}$$

Afterwards, we quantize it as follows:

$$(K_t)_n = \frac{1}{\sqrt{4\pi t}} e^{-\frac{n^2}{4t}} \tag{11}$$

$$(x * K_t)_m = \sum_{n=-\infty}^{\infty} x_n \frac{1}{\sqrt{4\pi t}} e^{-\frac{(m-n)^2}{4t}} \tag{12}$$

At this point, it is important to consider the relationship between the Gaussian filtering and the heat equation, given by $\partial u/\partial t = \partial^2 u/\partial x^2$, where $u(t,x)$ is the solution of the equation. Given a signal f, the result of convolving f with the Gaussian filter K_t is equivalent to the solution of the heat equation using f as the initial data $u(t,s) = K_t * f(x)$.

Considering this relationship, a discrete version of the heat equation can be used to accelerate the approximation of the Gaussian filtering (see [1] for more details), which results in a recursive scheme in three steps for each direction, as shown below, where I_0 is the original image:

$$\begin{array}{ll} I_j^{n+\frac{1}{3}} = I_j^n + vI_{j-1}^{n+\frac{1}{3}} & \forall j \in Z \\ I_j^{n+\frac{2}{3}} = I_j^{n+\frac{1}{3}} + vI_{j+1}^{n+\frac{2}{3}} & \forall j \in Z \\ I_j^{n+1} = \frac{v}{\lambda} I_j^{n+\frac{2}{3}} & \forall j \in Z \end{array} \quad (13)$$

This process will be performed by rows and by columns in order to obtain a discrete expression for a two-dimensional Gaussian filtering. Making use of the features of the Gaussian kernels, the result of applying a Gaussian filter with an initial scale t can be used to obtain a Gaussian filtering of the initial image for a different scale without needing to start again from the input. We will discretize the scale considering $\sigma_n = n\sigma_0$ for a given σ_0. Taking into account the relation $\sigma^2 = 2t$, the step size Δt to go from σ_n to σ_{n+1} is given by:

$$\Delta t = \frac{((n+1)\sigma_0)^2}{2} - \frac{(n\sigma_0)^2}{2} = \left(n + \frac{1}{2}\right)(\sigma_0)^2 \quad (14)$$

If we use $niter$ iterations of the recursive scheme in (13) to compute $I_{\sigma_{n+1}}$ from I_{σ_n}, the discretization scheme for the heat equation is given by:

$$\delta_t^{n+1} = \frac{\left(n+\frac{1}{2}\right)(\sigma_0)^2}{niter} \quad (15)$$

4.2 Scale Estimation

We must take into account that, for a certain texture, the use of different resolutions forces us to apply Gaussian functions with different standard deviations, thus requiring an adaptation stage. To do that, we first extract the evolution of the addition of the squares of the gradients at different scales, and then we use these factors to compare the textures. Even if the quantitative distribution of the orientations may be alike for different textures, the spatial distribution will cause a divergence in the evolution and interaction, so the factors will differ.

One of the properties of the Gaussian filtering is the relationship between the resolution of two images and the effects of this kind of filters. In fact, the result of applying a Gaussian filter with standard deviation σ to an image with resolution factor k is equivalent to applying a Gaussian filter with standard deviation $k\sigma$ to the same image acquired with a resolution factor kx.

Lemma 1 *Let $I_0(x,y)$, $I'_0(x,y)$ be such that there exists a constant k satisfying that $I'_0(x,y) = I_0(kx,ky)\ \forall (x,y) \in \Omega$, then $I'_t(x,y) = I_{k^2t}(kx,ky)$.*

Proof. The result follows from the uniqueness of the solution of the heat equation taking into account that the function $I_{k^2t}(kx,ky)$ is a solution of the heat equation for the initial datum $I'_0(x,y)$.

Given two textures, I_0 and I'_0, we will estimate the scale factor k using the normalized evolution of the norm of the gradient, that is, we will use:

$$\phi(I_0,\Omega,t) = \frac{\sqrt{\int\limits_\Omega |\bigtriangledown I_t|^2}}{\sqrt{\int\limits_\Omega |\bigtriangledown I_0|^2}} \quad (16)$$

It is well known (see for instance [3]) that $\phi(I_0,\Omega,t)$ is a decreasing function with respect to t and $Lim_{t\to\infty}\phi(I_0,\Omega,t) = 0$. On the other hand, from the previous lemma, we deduce that if $I'_0(x,y) = I_0(kx,ky)\ \forall (x,y) \in \Omega$ then:

$$\phi(I_0,\Omega,t) = \phi(I'_0,k\Omega,k^2t) = \phi(I'_0,\Omega,k^2t) \quad (17)$$

So in order to estimate a scale factor k between two textures I_0 and I'_0, we will compare the functions $\phi(I_0,\Omega,t)$ and $\phi(I'_0,\Omega,t)$. Let $r_n^1 = \phi(I_0,\Omega,(\sigma_n)^2/2)$ and $r_n^2 = \phi(I'_0,\Omega,(\sigma_n)^2/2)$ be the ratios obtained for two textures at the scale $\sigma_n = n\sigma_0$, the best adjusting coefficient k to fit the series of r_n^2 to that of r_n^1, both consisting of N terms, can be obtained as follows: First, we fit a value $0 < h < 1$ and we interpolate the values in the series r_n^1 and r_n^2 to obtain two new series σ_n^1 and σ_n^2 which estimate the scales for which the ratios $(1, 1-h, 1-2h, 1-3h, ..., 1-(N-1)h)$ are obtained. In other words, we estimate the scale where $\phi(I,\Omega,(\sigma_n^1)^2/2) = 1 - nh$.We point out that if $nh < 1$, σ_n^1 and σ_n^2 are well-defined because $\phi(I,\Omega,t)$ is a decreasing function with respect to t and $Lim_{t\to\infty}\phi(I_0,\Omega,t) = 0$. With these values, we minimize the following error to obtain the scale factor k:

$$e(k) = \frac{1}{N}\sum_{i=0}^{N-1}\left(\sigma_i^1 - k\sigma_i^2\right)^2 \quad (18)$$

$$\frac{de(k)}{dk} = 0 \implies k = \frac{\sum_{i=0}^{N-1}\left(\sigma_i^1\sigma_i^2\right)}{\sum_{i=0}^{N-1}\left(\sigma_i^2\right)^2} \quad (19)$$

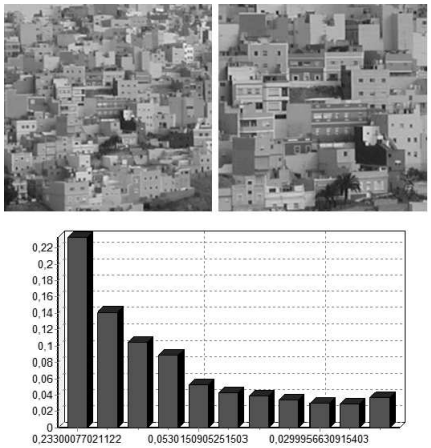


Figure 6: Comparison of two similar textures at different scales

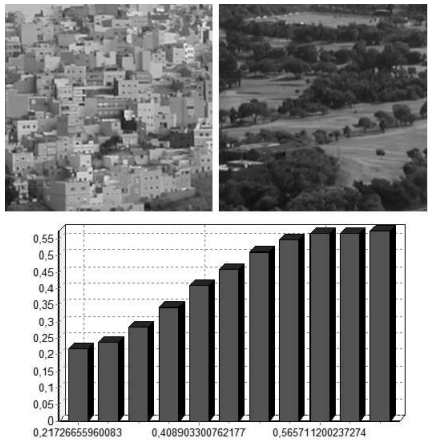


Figure 7: Comparison of two different textures at different scales

4.3 Multiscale Texture Orientation Histogram Comparison

We can study how the energy obtained when comparing the orientation histograms evolves as we apply a Gaussian filtering to the textures. We use the adjusting factor k, as in (19), to relate the scales to be compared. Finally, we obtain the energies for the comparison of the histograms at N different scales, given by (20), where $n = \{0, .., N-1\}$ and σ_N is the minimum of σ_N^1 and $k\sigma_N^2$.

$$\sigma_n = \frac{n}{N}\sigma_N \qquad \text{and} \qquad \sigma'_n = \frac{nk}{N}\sigma_N \tag{20}$$

Figure 6 shows the results of comparing two images of the database corresponding to similar textures, acquired at different distances. As observed, not only the initial energy is low, but also the subsequent energies, obtained when comparing the images at the corresponding scales, decrease when we increase the scale. On the other hand, Fig. 7 shows the comparison of two images of different textures and the energies, far from decreasing, increase from the initial value.

4.4 Ratio Comparison in Natural Textures

We have extracted the evolution of the square of the gradient across the image for all the textures in the second database, in which different scenes have been acquired at different distances. With these values, we have extracted a ratio for every comparison of two pictures in the database. Instead of observing a great variability in the ratios according to the different natures and distances, they are very close to 1 in most of the cases. The fact that certain particular elements appear when we approach them, while other global elements disappear, thus generating new gradients while other ones are eliminated, makes the total addition similar, and the information, in terms of changes present in the image, is approximately constant. In fact, the mean ratio for the comparison of two textures, considering in each case the ratio which is lower than 1, is 0.91975, with standard deviation 0.06191. In artificial textures, a change in the resolution produces a change in the evolution of the addition of the squares of the gradients and no additional information is added, thus generating more variable ratios.

5 Conclusion

In this work, we have presented a new approach to texture classification and a study on multiscale analysis of natural textures. By using the structure tensor, we have obtained an estimation of the gradient in every point of the textures. The extraction of orientation histograms to describe the distribution of the orientations across a textured region permits us to perform an initial clustering of the textures according to the quantitative and relative distribution of the different orientations.

The comparison of the Fourier coefficients and certain normalization processes which have been included allow a satisfactory classification in many cases, including size and rotational invariance. However, due to the ambiguities that are generated by the non-injectivity of the generation of these histograms, a further study has been carried out, by comparing the evolution of the histograms at different scales.

This multiscale analysis of the histograms has produced quite satisfactory results, since the visual similarity or dif-

ference between two textures is much more reliably detected by the evolution of the energies resulting when comparing the histograms at different scales, which have been previously adjusted. We have extracted the scale factor which must be used when comparing two textures to perform the comparison appropriately. We have observed how this ratio is not far from 1 when natural images are considered, since the information contained in them changes qualitatively, but not so much quantitatively.

The numerical results obtained in the tests which have been implemented confirm the usefulness of the multiple comparison of the images, since they endow us with a much more robust discrimination criterion. At the same time, the tests on natural images show the changes in the information contained in the different views of a scene.

References

[1] Alvarez, L., Mazorra, L.: Signal and image restoration using shock filters and anisotropic diffusion. SIAM J. on Numerical Analysis **31:2** (1994) 590-605.

[2] Columbia University and Utrecht University. Columbia-Utrecht Reflectance and Texture Database. http://www.cs.columbia.edu/CAVE/curet/.index.html.

[3] Evans, L.: Partial Differential Equations. American Mathematical Society (1998).

[4] Lindeberg, T.: Scale space theory in computer vision. Kluwer Academic Publishers (1994).

[5] Paragios, N., Deriche, R.: Geodesic active regions and level set methods for supervised texture segmentation. International Journal of Computer Vision **46:3** (2002) 223.

[6] Weickert, J.: Multiscale texture enhancement, V. Hlavac, R. Sara (Eds.), Computer analysis of images and patterns, Lecture Notes in Computer Science Springer Berlin **970** (1995) 230-237.

AMICam: A Video Processing Software for Combining Real and Virtual Scenes

L. Alvarez, C. Cuenca, A. Salgado and J. Sanchez
Departamento de Informática y Sistemas
Universidad de Las Palmas de Gran Canaria
Campus Universitario de Tafira, 35017 Las Palmas, SPAIN

Abstract

In this paper, we present a brief description of a master thesis done by Agustín Salgado in the computer science department of Las Palmas University. The master thesis deals with the problem of inclusion of $3-D$ virtual objects in a real video sequence. We do not introduce any new real relevant contribution in the field, we focus our attention in the hole problem and we try to provide a solution for the different tasks we have to deal with. In particular, we study techniques for characteristic point extraction, tracking, multiple camera calibration, synchronization of real and virtual cameras and the rendering of virtual objects in the real video sequence. Finally, we present the experimental results obtained. In particular, we insert some $3-D$ virtual objects in a real video sequence and we show the result obtained and the problems we have noticed.

1. Introduction

The digital image technology has experimented an important increase for the last years, mainly, due to the increase of computers performance and the digital video. This technology allows the combination of synthetic objects and real scenes. We can add virtual objects to real video sequences in such a manner as to appear part of the 3D world. This technology can be applied in many fields, like movies or advertisement. This is the context of the master thesis we present here.

In this work, we use a real video sequence recorded with a video camera which has been moved thought a 3D scene. We want to include one or various virtual 3D objects. After that, a new video sequence will be created with the original sequence and the 3D objects. The final video sequence will be equivalent that the virtual objects would have been in the scene when the video sequence was recorded.

The organisation of the paper is as follows: In section 2, we introduce the Harris corner detector technique that we use to extract characteristic points in an image. In section 3, we study the problem of tracking the characteristic points, obtained in each image by the Harris detector, across the video sequence. In section 4, we present an overview of multiple camera calibration techniques. In section 5, we study the render process to generate new video sequences from the inclusion of virtual 3D objects in a real video sequence. In section 6, we present a video processing software for combining real and virtual scenes. Finally, in section 7 and 8, we present the experimental results and the main conclusions of the paper.

2. Harris corner detector

An image contains information about a scene. However, only a few of this information let's us to understand the scene. We will try to extract that information that help us to know the camera movement. A corner is a kind of information it presents in many images and it is useful for the camera calibration.

In this section, we focus in the Harris corner detector algorithm develop by AMI group. This corner detector has two important features, it is very fast and the corners information is precise. This implementation can estimate the corner position in subpixel precision by interpolating the Harris values and computing the maxima of some interpolated function.

We tested the Harris detector on real images. The number of corners detected depends directly of the three parameters of Harris detector. The behaviour of this corner detector is determined by three parameters, which are described as follows:

- *Harris radio.* It shows the minimum distance (in pixels) allowed between two corners. When the detection process finishes, the best corners are chosen by spacial location.
- *Sigma.* It shows the standard deviation of the Gaussian. It is used to balance the derivates values around a pixel.
- *Threshold.* It is the Harris value threshold used to choose the pixels which are corners. It affects directly the number of corners detected.

When we select the value of the Harris detector parameters, we must be a deal, between the number of corners

detected and the total computational time. Restricted values reduce the computational time. If the number of corners is high, it will affect to the total computational time of the whole video creation process. The experiment results have shown that the corner method has good detection and excellent location performance.

3. Tracking sequences of corresponding singular points across the video sequence

In this section, we describe the tracking process. This process computes automatically sequences of corresponding singular points across the video sequence. The path of a 3D point is a list of projection points in following frames, where that 3D point is shown.

3.1. Succession criterias

The tracking computes sequences of corresponding singular points across the video sequence. The corners detected are local information in each frame. So, we must try to relation the corners in following frames. The tracking process needs some criterias to know when two corners are in correspondence. We define two criterias: Harris and Correlation test. These criterias are the tracking "intelligence".

The Harris detector gives us for each corner, its location (x, y) in subpixel precision and its Harris value. If two corners, placed in following frames, correspond to the same 3D point they will have similar Harris values. The Harris test compares the Harris values of two corners. The second test, correlation test, computes the correlation between two windows centred in two corners. These tests are passed when the comparation and correlation results are less than a threshold.

3.2. Tracking process description

The tracking is an iterative process. Its aim is to compute the path of the singular points across a video sequence. This process is divided into the following steps:

Step 1:

In the first step we extract the singular points from all frames of the video sequence. We use the Harris detector described in the previous section.

Step 2:

It is a simplification of the tracking process. We have only two frames and we want to know the correspondence of a set projections in these frames. If we want to extrapolate this solution to the rest of the sequence, we consider in the next iteration, the second frame first one, and the following (to that second frame) second one.

Now, for each corner, we try to find its predecessor point in the previous frame.

1. Looking for predecessor.

   Fixed a singular point, we look for its predecessor point into a window NxN, centred in the corner location, in the previous frame. We will go to the next step, only if we find a candidate predecessor point. Otherwise, we consider this singular point the beginning of a new sequence.

2. Apply the Harris and correlation test.

   In this step, we apply the succession criterias. If the two tests are passed, we consider these singular points in correspondence. When one of these tests is not passed, we came back the previous step and the search continues.

Corre Har	**15%**	**30%**	**50%**
5%	363.825	332.989	311.059
10%	345.129	300.762	266.779
20%	330.398	275.143	230.590

Table 1: Number of sequences obtained.

3.3. Tracking process experimental results

In this part, we examine the performance of the tracking process. We tested the tracking process on a real video sequence of 171 frames, where an average 1200 corners per frame was detected. This process was tested and compared on their basis tests: *Harris* and *Correlation*. In figure 1, we can see the graphical results obtained. In table 1, we can see the results obtained when we change the thresholds values for the Harris and Correlation test.

The Harris test avoids to apply the Correlation to non-correspondence corners. So, it reduces the compute time. However, the thresholds must not be restricted, because the sequences length will be short. Short sequences are not useful for the camera calibration.

With the same way it happened in the previous section, we have to get a deal between restricted and tolerated threshold values, to obtain good results in short time.

4. Camera Calibration

The problem of multiple camera calibration consists in recovering the camera positions and orientations with respect to the world coordinate system, using as input data tokens,

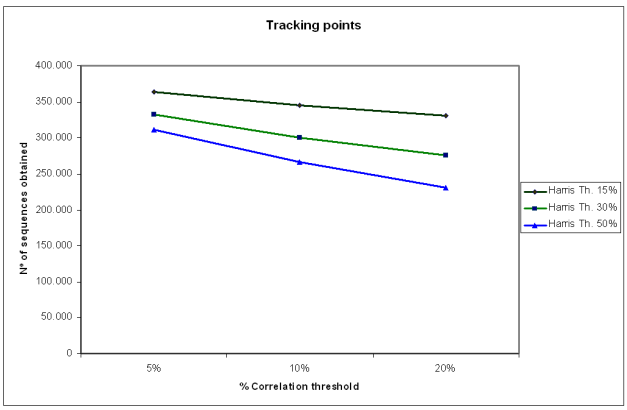


Figure 1: Graphical results of the table 1. It shows the numbers of sequences obtained when we modify the Harris and Correlation thresholds.

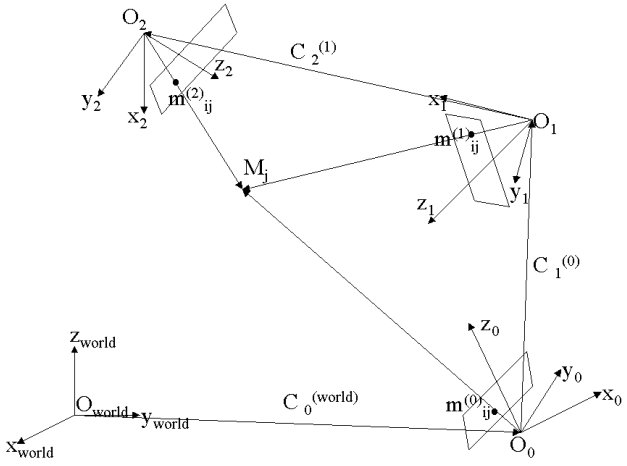


Figure 2: Motion parameters derived from point matches.

such as pixels or lines, in correspondence in different images. Figure 2 shows this scenario for a system with three cameras.

The specification of the i-th camera position is the 3D point $C_i^{(world)}$, where the superscript is the reference system in which the magnitude is expressed. The orientation specification is a rotation matrix $R_i^{(world)}$ or any equivalent representation, such as quaternions or Euler angles.

When the image tokens in correspondence are projections of a set of 3D points $\{M_j\}_{j=1..N}$ where N is the number of points, it is possible to reconstruct each 3D point expressed in the world coordinate system by simply estimating the intersection point of the line set:

$$\left\{ r_i \equiv C_i^{(world)} + \lambda \overrightarrow{C_i^{(world)} R_i^{(world)} m_{ij}^{(i)}} \right\}_{i=1..N}$$

where $C_i^{(world)}$ are the coordinates of the optical center in the world reference system, and $m_{ij}^{(i)}$ are the coordinates of the projection of M_j in the normalised reference system for the i-th camera. A reference system is said to be normalised when the optical center is in the origin, the focal distance is 1 and the pixel is a square of size 1. We will assume that the intrinsic parameters of the cameras are known, which allows us to normalise the reference system.

In order to estimate the intersection 3D point of the line set it is necessary to know the position of the optical center and the rotation matrix for each one of the cameras. The computation of these parameters solves the problem of the multiple camera calibration. After estimating these parameters, we can evaluate the accuracy of the solution by projecting the reconstructed 3D points in each camera, and the best solution for the calibration problem is the one that minimises the energy function:

$$f(C_0^{(world)}, C_1^{(world)} \quad , \quad ..., R_0^{(world)}, R_1^{(world)} ...) = \sum_{i,j} \left\| m_{ij}^{\prime(i)} - m_{ij}^{(i)} \right\|^2$$

where $m_{ij}^{\prime(i)}$ is the projection of the reconstructed point M_j in the i-th camera.

To study the problem of camera calibration, we will use the classical "pinhole model" which assumes the simplest projective model for the camera image acquisition. To calibrate the camera means to find out the parameters which determine the way that the projection works. There are two types of parameters: intrinsic and extrinsic parameters. The extrinsic parameters determine the $3-D$ location of the camera in the scene with respect to some *a priori* fixed reference system. They are a translation vector and rotation matrix. The intrinsic parameters do not depend on the camera location in the $3-D$ scene. They are dependent of the camera, and are the focal length, the pixel size and the focus position. A camera is defined by the projection matrix. This matrix determines the projection from the 3D space (world points) into 2D space (projection plane).

The calibration algorithm returns one projection matrix for each frame. We use this information to position the camera into the scene.

5. Inclusion a virtual object in the video sequence, render and creation of a new video sequence

In this section, we describe the inclusion of virtual $3-D$ objects process into a scene and the making of a new video sequence. We have two types of cameras, a real and other

artificial, designed for solving problems completely different. Our aim is the projections of both cameras will be identical. If you get to fit the projections, the final video will be equivalent that the virtual objects would have been in the scene when the video sequence was recorded.

5.1. Inclusion a virtual object in the video sequence

To simule the inclusion a 3D virtual object in a real video sequence, we make a 3D world and the $3-D$ objects are included into it. In our virtual 3D world, the user can modify the $3-D$ objects, as their location as the transformations to apply them. To complete our 3D world we put in the background a frame (extracted from the video sequence).

To make the 3D world we use a graphical library, such as Open Inventor. Open Inventor is a library of objects and methods used to create interactive 3D graphics applications.

5.2. Render process

This process is automatically computed by Open Inventor render engine, so we avoid the complexity of this process.

5.3. Creation the new video sequence process

When the $3D$ objects are placed into the scene, we create the new video sequence with the virtual objects inserted. To keep the sense the objects are in the real-world scene, we have to put in the background the frame that is watched from the camera location. This process is divided into the following steps, which are executed for each frame:

1. We create a new frame, which size is the same that the original. We copy the original frame on a new one.
2. The camera is positioned in the scene, with the information extracted from the projection matrix of this frame. After that, Inventor renders our scene (only objects) and its output is stored in a buffer.
3. Finally, we join the frame (background) and the render output. We overwrite on the frame those pixels that correspond with the objects projection.

5.4. Experimental results

When the virtual objects are inserted in the scene, we assume they stay static. This master thesis does not consider dynamic objects. To keep the static objects sense, the projections of the real and Inventor cameras must be identical. However, the real and Inventor cameras have been designed for solving problems completely different and they have some difference.

In one hand, the real camera has intrinsic and extrinsic parameters. The extrinsic parameters determine the $3-D$ location of the camera in the scene with respect to some *a priori* fixed reference system. The intrinsic parameters do not depend on the camera location in the $3-D$ scene. They are dependent of the camera, and are the focal length, the pixel size and the focus position. In the other hand, the Inventor camera projects objects in a 3D virtual scene. So, this camera has extrinsic parameters but it has not intrinsic parameters (or it has ideal intrinsic parameters).

Camera		(X_1, Y_1)	(X_2, Y_2)
1	P_{2D}	(163.50, 888.07)	(293.78, 987.6)
	P_{IV2D}	(163.25, 888.07)	(293.59, 987.6)
	Dif	(0.25, 0.0)	(0.19, 0.0)
50	P_{2D}	(336.63, 889.75)	(469.28, 988.31)
	P_{IV2D}	(336.47, 889.75)	(469.18, 988.31)
	Dif	(0.16, 0.0)	(0.10, 0.0)
100	P_{2D}	(551.16, 889.34)	(686.21, 983.86)
	P_{IV2D}	(551.10, 889.34)	(686.22, 983.86)
	Dif	(0.06, 0.0)	(-0.01, 0.0)

Table 2: Projection comparative of a 3D point in the real and Inventor cameras.

To patch the difference between the intrinsic parameters of both cameras, we assume in the real camera that the focus projection is in the center of the projection plane and the pixel is square (the same width and height). In table 2, we can see the experimental results obtained, when we placed two spheres in the corner location in the real-world.

6. AMICam

One of this master thesis aims was the design and building of an application. This application is called *AMICam*. AMICam is an application that offers to users tools for inclusion of $3-D$ objects in a video sequence. The tools are easy to use for any user.

Our application is divided in two parts: the interface and compute programs. The interface manages and stores the information the user inserts in the system. The compute programs are in a lower level, and they computes the most important process of this master thesis (tracking, camera calibration and render). An AMICam snapshot is shown in figure 3.

7. Experimental results. Inclusion of virtual objects in a real video sequence

One of the main applications of video sequence calibration is the inclusion of virtual $3-D$ objects in a real video sequence. We tested our methods in a real video sequence of

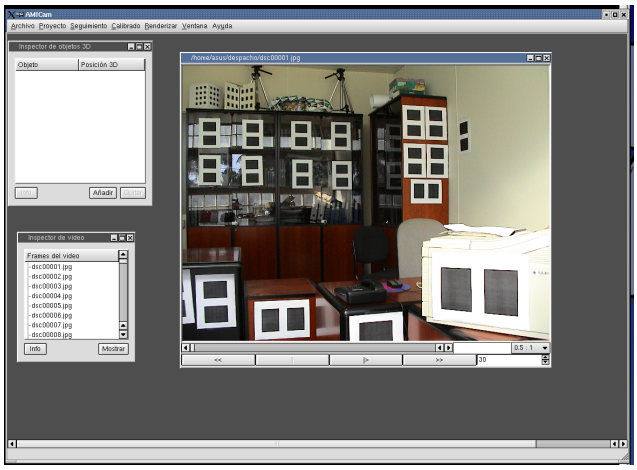


Figure 3: Main window of AMICam.

120 frames where we included eight artificial objects. The video sequence was taken with a digital camera. The camera zoom was constant and the camera was placed in the end of the office.

We have used AMICam to create the new video sequence with the virtual $3-D$ objects inserted. We placed in the office some white sheets with black squares printed on them. The black squares corners are easily detected by Harris corner detector. The points sequences detected by tracking process were long (average of 70 points per sequence, 120 frames). When the tracking finished, we selected the best sequences for the calibration process.

We have inserted eight virtual $3-D$ objects, in different planes and depth. In table 3 and 4, we present six frames of the real video sequence (left column) and the same frames with the inclusion of the virtual objects using the calibration parameters obtained (right column).

The final results are very well, the objects stay in their location. However, it exists an error between half and one pixel. This results could be improved if we use a nonlinear algorithm, like Levenberg-Maquard.

8. Conclusions

In this paper, we present a master thesis oriented to inclusion $3-D$ object in a video sequence. In this field, the techniques must be, efficiency, robustness, precise and flexibility. We have focus on reach these aims: (1) development and implementation of methods for inclusion $3-D$ objects in a video sequence and (2) development of a user interface that manages those methods (AMICam).

The final results, how we could see, were excellent and the range of application of the proposed methods are very wide.

References

[1] Faugeras, Olivier, *Three-Dimensional Computer Vision* The MIT Press. (1993).

[2] Richard Hartley and Andrew Zisserman, *Multiple View Geometry in Computer Vision* The MIT Press. (2001).

[3] Alvarez, Luis and Cuenca, Carmelo and Mazorra, Luis, *Signal Processing, Pattern Recognition and Applications. IASTED.* Morphological Corner Detector. Application to Camera Calibration. (2001).

[4] Roger S. Pressman, *Ingeniería del Software, quinta edición* McGraw-Hill Companies. (2001).

[5] Josie Wernecke, *The Inventor Mentor* Addison Wesley Publishing Company. (1994).

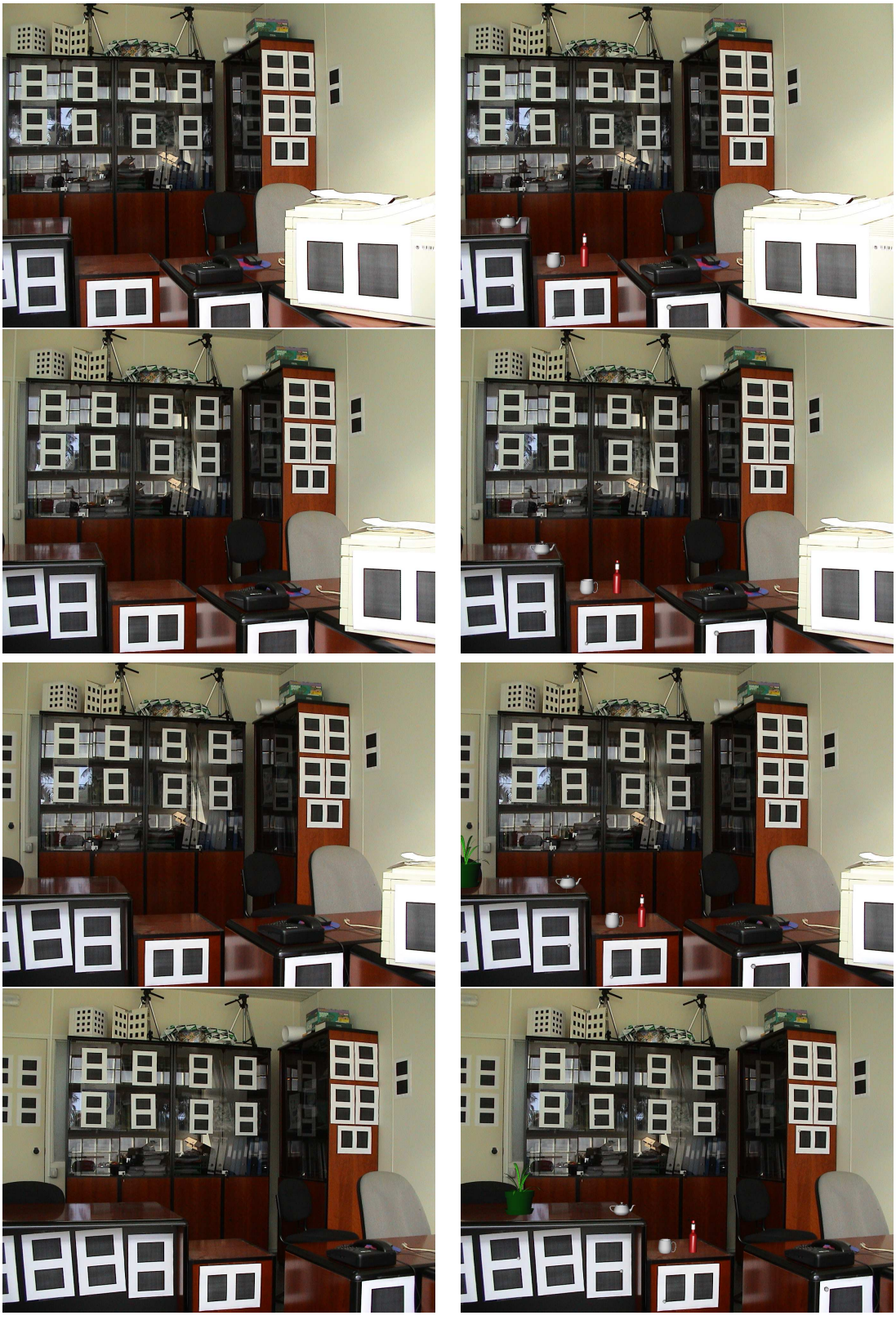


Table 3: On the left, four frames (1, 30, 51 and 91) of a real video sequence and on the right the same frames with the inclusion of the virtual objects (a red bottle, a mug, a teapot and a plant).

Table 4: On the left, two frames (101 and 120) of a real video sequence and on the right the same frames with the inclusion of the virtual objects (a red bottle, a mug, a teapot and a plant).

Instituto Universitario de Ciencias y Tecnologías Cibernéticas
Universidad de Las Palmas de Gran Canaria
Campus de Tafira
35017 Las Palmas, España
http://www.iuctc.ulpgc.es