Real-time estimation of dominant motion in underwater video images for dynamic positioning

Fabien Spindler, Patrick Bouthemy
IRISA / INRIA Rennes
Campus universitaire de Beaulieu, 35042 Rennes Cedex, France
Email {fspindle, bouthemy}@irisa.fr

Abstract

In this paper, we propose a 2D visual motion estimation method which can be exploited to achieve a dynamic positioning (eg. by gaze control) with respect to a sea-bottom area of interest of a video camera mounted on a subsea vehicle. It mainly involves a dominant 2D motion robust estimation step from underwater video sequences. Optimizations carried out on the motion estimation code have made possible the use of our algorithm in "application-related real-time" for scientific exploration or inspection tasks. We have developed a friendly and efficient interface to perform this algorithm in an operational context. Experiments dealing with complex real underwater scenes are reported and validate the approach.

1 Introduction

In the context of subsea robotics, video has already been exploited in previous works, for navigation [1], trajectography [3], stabilization [2], or motion estimation [4] tasks. We aim at performing video-based dynamic positioning of an underwater vehicle for exploration or inspection tasks. More precisely, we are interested in providing the required motion information extracted from images of a camera mounted on a subsea vehicle, to realize the fixation of this camera to a part of interest of the sea-bottom in scientific exploration, or to a given object in inspection tasks.

To achieve the dynamic positioning, we propose to estimate the apparent 2D motion induced in the image sequence by the subsea vehicle movement, in order to compensate for it. However, the possible presence of secondary motions, the low quality of underwater video images, and the usual subsea image contents make difficult such a motion estimation. This is why the use of a dominant motion robust estimator based on a multiresolution framework [7] was considered.

To perform a vision-based task in the underwater robotics field, it is usually required that the vision algorithm runs at least at the rate of 2 Hz. Thus, to be able to retrieve the dominant motion estimation in "application-related real-time", optimizations of the corresponding code were carried out. An interface supplying a friendly and efficient way to specify and to use the algorithm in an operational context has also been developed.

The next section of this paper recalls the main features of our dominant motion estimation method. Section 3 describes optimizations realized to run the motion estimation algorithm in real-time, and presents the interface used to exploit the dominant motion algorithm. Section 4 contains experimental results obtained with real underwater video sequences. Finally, concluding remarks are given in Section 5.

2 Dominant motion estimation

The camera-related image motion estimation process will involve data taken over the whole image, and then must be robust to the presence of moving elements or other disturbing spatio-temporal phenomena. In order to consider the dominant motion model resulting from this estimation step as being the 2D motion due to camera movement, we must assume that the projections of the static components of the scene occupy the main part of the image (with the additional condition that they supply image spatial intensity gradient information).

Usually, the choice of a motion model depends on considerations related to the kind of 3D motion undergone by the camera and objects, the type of transformation sufficient to account for the projection of the scene into the image plane, and the analytic description of the viewed surfaces. The use of a full 3D model (3D motion and depth) in the analysis process would lead to solve the general -but highly complex- problem of 3D reconstruction in the presence of moving objects. We prefer to use 2D parametric motion models $\vec{w}_\Theta$ to represent the projection of the 3D motion field of the static background, where $\vec{w}_\Theta$ denotes the modeled velocity vector field and Θ the set of model parameters. Such models are globally valid when either the camera translation magnitude is small with respect to the depth of the objects, or when there is not too much depth variation in the scene. Though less general than the full 3D case, the choice of 2D models leads to an efficient motion computation.

Among different possibilities, the 2D affine motion model proved to be a good compromise between its relevance as a motion descriptor and the efficiency of its estimation. It is defined at pixel $\mathrm{p} = (x, y)$ by:

$$\vec{w}_\Theta(\mathrm{p}) = \begin{pmatrix} a_1 + a_3 x + a_4 y \\ a_2 + a_5 x + a_6 y \end{pmatrix}$$

where $\Theta = (a_i), i = 1..6$, is the parameter vector to be estimated. However, when the background is approximately located in a plane whose slant is important, a particular quadratic motion model should be used.

To estimate the dominant motion model between two successive images I_t and I_{t+1}, we have developed a gradient-based multiresolution robust estimation method described in [7]. It can be applied to any kind of 2D polynomial motion models. To ensure the goal of robustness, we minimize an M-estimator criterion with a hard-redescending function [6]. The constraint is given by the usual assumption of brightness constancy of a projected surface element over its 2D trajectory [5]. Thus, the motion model estimation is defined as:

$$\begin{aligned} \widehat{\Theta} = \underset{\Theta}{\text{argmin}}\ E(\Theta) &= \underset{\Theta}{\text{argmin}} \sum_{\text{p}\in R(t)} \rho\left(\text{DFD}_\Theta(\text{p})\right) \\ \textit{with}\quad \text{DFD}_\Theta(\text{p}) &= I_{t+1}(\text{p} + \vec{w}_\Theta(\text{p})) - I_t(\text{p})\,. \end{aligned}$$

$\rho(x)$ is a function which is bounded for high values of x. More precisely, we use Tukey's biweight function. The estimation support $R(t)$ usually consists of the whole image. If required, it can also be restricted to a specific area of the image. The minimization takes advantage of a multiresolution framework and of an incremental scheme based on the Gauss-Newton method. At each incremental step k (at a given resolution level, or from a resolution level to a finer one), we have: $\Theta = \widehat{\Theta}_k + \Delta\Theta_k$, where $\widehat{\Theta}_k$ is the current estimate of the parameter vector Θ. A spatial linearization of $\text{DFD}_\Theta(\text{p})$ around $\widehat{\Theta}_k$ is performed, leading to a residual quantity $r_{\Delta\Theta_k}(\text{p})$ linear with respect to $\Delta\Theta_k$:

$$\begin{aligned} r_{\Delta\Theta_k}(\text{p}) \;\; = \;\; & \vec{\nabla} I_t(\text{p} + \vec{w}_{\widehat{\Theta}_k}(\text{p})).\vec{w}_{\Delta\Theta_k}(\text{p}) \\ & + I_{t+1}(\text{p} + \vec{w}_{\widehat{\Theta}_k}(\text{p})) - I_t(\text{p}) \end{aligned}$$

where $\vec{\nabla} I_t(\text{p})$ denotes the spatial gradient of the intensity function at location p and at time t. Then, we substitute for the minimization of $E(\Theta_k)$ the minimization of the expression given by $E_a(\Delta\Theta_k) = \sum \rho(r_{\Delta\Theta_k}(\text{p}))$. This error function is minimized using an Iterative-Reweighted-Least-Squares procedure, with 0 as an initial value for $\Delta\Theta_k$. For more details about the method (called RMR Method) and its performances, the reader is referred to [7].
This estimation algorithm allows us to get a robust and accurate estimation of the dominant motion model (i.e., background apparent motion) between two images. It will be used to realize the dynamic positioning of the camera to the region of interest by compensating for the 2D dominant motion induced by the camera movement.

3 Optimization and interfacing of the motion estimation algorithm

In order that the dominant motion estimation algorithm runs in "application-related real-time" on an ultra-Sparc-1 Sun workstation equipped with a SunVideo image capture board, we have carried out substantial optimization steps. A Tcl/Tk interface supplying a friendly and efficient way to specify and to use the algorithm has also been developed.

3.1 The algorithm optimization

After analyzing the profile specifying the execution time for each routine of the motion estimation program, we have optimized fifteen routines requiring about 95 % of the CPU. This has involved :

- the conversion from double to float in all the computations. On DSP or RISC processors, a float calculation is actually 30 % faster than a double one. Moreover, float data structures need half memory space than double one. We have checked that the precision loss induced by this conversion has no perceptible effect on the motion estimation performances.
- the reduction of elementary computation operations (addition, multiplication and assignment operations). The following example illustrates this optimization step. Let us consider the case of a symmetric filter to be applied to the image values. Let us note : N, the filter size with $N = 3, 5$ or 7 ; c_i, the filter coefficients with $c_i = c_{N-1-i}$ for $i \leq N/2$; l_i, the pixel intensity and l, the filtered pixel intensity. The initial form of the symmetric filter is given by :

  $$l = \sum_{i=0}^{N-1} c_i * l_i$$

  This leads to $N-1$ additions and N multiplications. If we consider the modified version as follows :

  $$l = c_{N/2} * l_{N/2} + \sum_{i=0}^{N/2-1} c_i * (l_i + l_{N-1-i})$$

  this leads to $N-1$ additions and $N/2+1$ multiplications.
- the monodimensional scanning of bidimensional data structures. Most of the software routines are scanning frames from top to bottom and from left to right, which amounts to a linear dependence of the memory space. In C language and for most of the C compilers, scanning of a memory space is better performed by pointers than by indexed memory addressing. With the appropriate control loop structure, the pointer scanning is 25 % faster.
- the manual loop unrolling. We have applied a manual loop unrolling optimization. It consists in a duplication of the internal loop instructions. It permits to better utilize the processor pipeline and to minimize the cost of the loop counter control. For RISC processors, the loop unrolling is part of the Gnu compiler options, eg. `-funroll-loop`. Thus, the performance of the manual loop unrolling is higher than the compiler one.
- the minimization of temporary data structures. The goal of the minimization, and even the elimination of temporary data structures, is to reduce the memory space used by the estimation software.

All the introduced optimizations were tested and validated on a lot of underwater video sequences. For images of size 256x256 pixels, with four levels in the multiresolution framework, the acceleration factor given by these software optimizations is about two. Table 1 gives the distribution of

Algorithm steps	Acceleration factor
pyramid construction	3
robust estimation	1,6
motion compensation	1,3
total	2

Table 1: Distribution of the acceleration factor in the dominant motion estimation algorithm, with images of 256x256 pixels and four levels in the multiresolution framework.

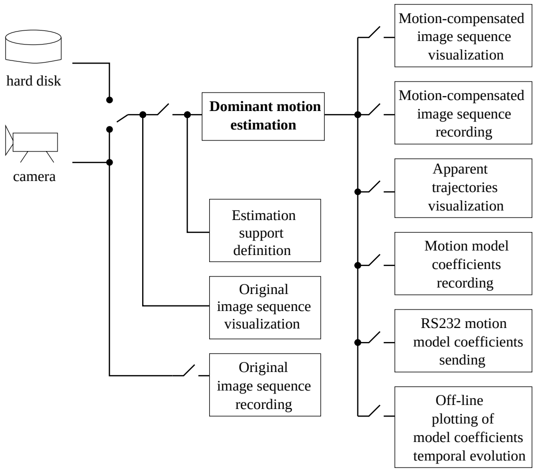

hard disk
camera
Dominant motion estimation
Estimation support definition
Original image sequence visualization
Original image sequence recording
Motion-compensated image sequence visualization
Motion-compensated image sequence recording
Apparent trajectories visualization
Motion model coefficients recording
RS232 motion model coefficients sending
Off-line plotting of model coefficients temporal evolution

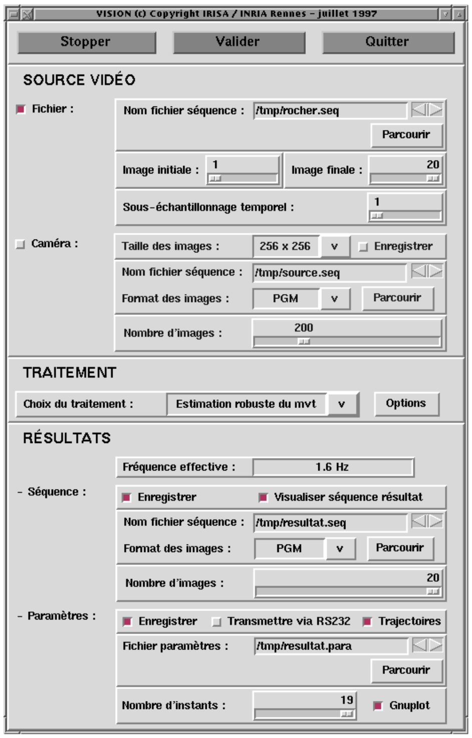
Figure 1: The interface functions.

the acceleration factor according to the motion estimation algorithm steps.

On an ultra-Sparc-1, these optimizations have allowed us to supply a dominant motion estimation result for images of 256x256 pixels at the rate of $1,6$ Hz. Moreover, if the estimation does not exploit the finest resolution level in the multiresolution framework, or similarly, if the estimation support takes into account a reduced area of the image, it is possible to reach a processing rate greater than 2 Hz. Let us point out that this CPU time values include the processing time of the motion-compensated image sequence construction step.

3.2 The interface

The main goal of the interface is to specify and to control the dominant motion optimized estimation process for underwater video sequences acquired by the SunVideo board connected to the camera. Moreover, the interface enables to store a video sequence acquired at video rate on hard disks. This allows the user to successively run the motion estimation on the same test sequence, in order to compare some options or to correctly set parameters. They involve the number of coefficients Θ used in the selected motion model (2, 3, 4, 6, 7, 8, 12 or 13), the first and last image pyramid levels considered in the multiresolution framework, the position and size of the estimation support. The above mentioned motion models respectively correspond to constant motion (translation), affine models and quadratic models, with the possibility to add an extraneous parameter to account for global illumination variation.


VISION (c) Copyright IRISA / INRIA Rennes – juillet 1997

SOURCE VIDÉO

☒ Fichier :
Nom fichier séquence : /tmp/rocher.seq

Image initiale : 1
Image finale : 20
Sous-échantillonnage temporel : 1

□ Caméra :
Taille des images : 256 x 256 v
□ Enregistrer
Nom fichier séquence : /tmp/source.seq
Format des images : PGM v

Nombre d’images : 200

TRAITEMENT

Choix du traitement : Estimation robuste du mvt v

RÉSULTATS

Fréquence effective : 1.6 Hz

- Séquence :
☒ Enregistrer ☒ Visualiser séquence résultat
Nom fichier séquence : /tmp/resultat.seq
Format des images : PGM v

Nombre d’images : 20

- Paramètres :
☒ Enregistrer □ Transmettre via RS232 ☒ Trajectoires
Fichier paramètres : /tmp/resultat.para

Nombre d’instants : 19
☒ Gnuplot

Figure 2: Tcl/Tk interface main window. Notice that the estimation software written in C language was compiled with `gcc -Wall -O2 -mv8 -msupersparc`.

To evaluate the algorithm performance and the validity of the estimated motion model, the interface supplies an on-line visualization at the estimation execution rate of the compensated image sequence with respect to a reference instant. This sequence is build by using the combination of the instantaneous dominant motion estimations between two successive frames. Moreover, the on-line incremental visualization of apparent trajectories of some points specified by the user is provided. It permits to qualify the apparent global motion. Finally, the temporal evolution of the estimated motion model coefficients can be viewed at the end of the estimation processing by plotting the corresponding graphics. Figure 1 summarizes all the interface functions, and Fig. 2 presents the main Tcl/Tk interface window.

4 Experimental results

In the experiments, the interface is used to estimate the dominant 2D affine motion model between two successive images of the underwater video sequences. These sequences were first acquired at video rate. Four level are used in the multiresolution framework. The estimated dominant motion model is than exploited to compute the compensated video sequence, in order to assess the performance of the dominant motion estimation algorithm, and to demonstrate that it will be possible to realize fixation on an interesting part of the sea-bottom by gaze control. This sequence is composed of frames warped to the first image of the original sequence.

Since $\vec{w}_{\widehat{\Theta}}(\mathrm{p})$ is generally a float expression, the warping step includes bilinear intensity interpolations. If the region of interest appears static in the compensated sequence, then the aimed task is satisfied.

4.1 The "rock" video sequence

Figures 3a, 3c and 3e show three images of the underwater "rock" video sequence. The dominant displacement in the image is induced by the camera moving around an hydrothermal chimney which can be considered as the object of interest in this scene. A flowing or a gas "cloud" leaks from the top of the rock creating a secondary motion. Actually, the dominant motion model well corresponds to the apparent motion of the rock. Once the dominant motion compensation step is performed, the fixation appears on the rock. This example is particularly complex because of the disturbing motion induced by the "cloud", and of the important depth effect on the rock which is not perfectly taken into account by the 2D affine motion model.

Figures 3b, 3d and 3f show that the algorithm realizes a proper estimation of the apparent motion of the rock during all the video sequence. The hydrothermal chimney remains static in the compensated sequence. Let us point out that these results computed at the rate of $1,6$ Hz could be more easily appreciated by visualizing the video sequence on the ultra-Sparc screen. When stopping the dominant motion model estimation on the last but one level in the multiresolution framework, we obtain a motion estimation rate of $2,3$ Hz. However, this gain is obtained to the prejudice of a slight decrease of the estimated motion model accuracy.

	a_1	a_2	a_3	a_4	a_5	a_6
t_1	-0,95	-0,07	0,003	0,002	0,001	0,002
t_{13}	-1,27	0,02	0,003	-0,001	0,000	0,001
t_{18}	-1,09	-0,00	0,004	0,002	0,000	0,001

Table 2: Estimated coefficients of the affine motion model at time t_1, t_{13} and t_{18} for the "rock" sequence.

Table 2 gives the values of the six coefficients of the estimated 2D affine motion model at time t_1, t_{13} and t_{18} for the "rock" sequence. The complete temporal evolution of the motion parameters over the processing sequence is given on Fig. 4.

4.2 The "zoom" sequence

Figures 5a, 5c and 5e show three images of the underwater "zoom" video sequence. The dominant displacement in the image is induced by camera zooming out on a octopus. Since the background of the scene corresponds to the main part of the image, the dominant 2D motion model correctly accounts for the apparent motion of the background.

Figures 5b, 5d and 5f demonstrate that the algorithm delivers a quite satisfactory estimation of the apparent motion of the background over all the video sequence. The rock located under the octopus and the white area on the left of the octopus correctly appears as fixed in the compensated sequence. These results were supplied at the rate of $1,6$ Hz. If we stop the dominant motion model estimation on the last but one level in the multiresolution framework, the processing rate grows up to $2,6$ Hz.

Table 3 gives the values of the six coefficients of the estimated 2D affine motion model at time t_1, t_{13} and t_{18} for the "zoom" sequence. The temporal evolution of these coefficients is plotted on Fig. 6.

	a_1	a_2	a_3	a_4	a_5	a_6
t_1	0.85	-0.71	0.001	-0.003	-0.002	0.000
t_{13}	-0.37	-0.91	-0.002	0.001	0.000	-0.003
t_{18}	0.37	-0.97	0.000	0.002	0.000	-0.001

Table 3: Estimated coefficients of the affine motion model at time t_1, t_{13} and t_{18} for the "zoom" sequence.

5 Conclusion

We have proposed an original solution to achieve a dynamic positioning of a subsea vehicle camera using a real-time estimation of the dominant motion in underwater video images. It has required the optimization and the interfacing of the 2D motion estimation algorithm we had originally devised.

In spite of the presence of secondary motions, the low quality of underwater video images, and depth effects, we are able to correctly estimate the global image motion related to the camera movement. This was demonstrated by compensating for this estimated dominant motion, and by supplying an appropriate fixation on the region of interest in the observed sea-bottom scene. Satisfactory results have been obtained on real video sequences depicting complex scenes, which validates our approach.

This work can be of practical interest for scientific exploration or inspection tasks in underwater robotics. In collaboration with Ifremer Toulon, the system described in this paper has been implemented in a subsea vehicle. Besides, we have effectively exploited the estimated dominant image motion parameters to achieve, in real-time, gaze control of a robot camera within a visual-servoing scheme.

Acknowledgments

This work was supported by Ifremer Toulon, France.

References

[1] J.-M. Aguirre, F. Boucher and J.P. Hue. Passive navigation of a submersible vehicle by image processing. *Proc. 4th European Signal Processing Conf., Grenoble*, pages 963–966, Nov. 1988.

[2] M. Amat, J. la Heras and R. Villa. Vision based underwater robot stabilization. In *2nd Workshop on Mobile Robots for Subsea Environments*, Monteray, USA, May 1994.

[3] P. Bouthemy and P. Lalande. Determination of apparent mobile areas in an image sequence for underwater robot navigation. *IAPR Workshop on Computer Vision: Special Hardware and Industrial Applications, Tokyo*, pp. 409–412, Oct. 1988.

[4] R. Caccia, M. Cristi and G. Veruggio. Motion estimation and modeling of the environment for underwater vehicles. *IARP Workshop on Underwater Robotics*, Toulon, March 1996.

[5] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, Vol.17:185–203, 1981.

[6] P.J. Hubert. *Robust statistics.* Wiley, 1981.

[7] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Jour. of Vis. Comm. and Im. Repr.*, 6(4):348–365, Dec. 1995.

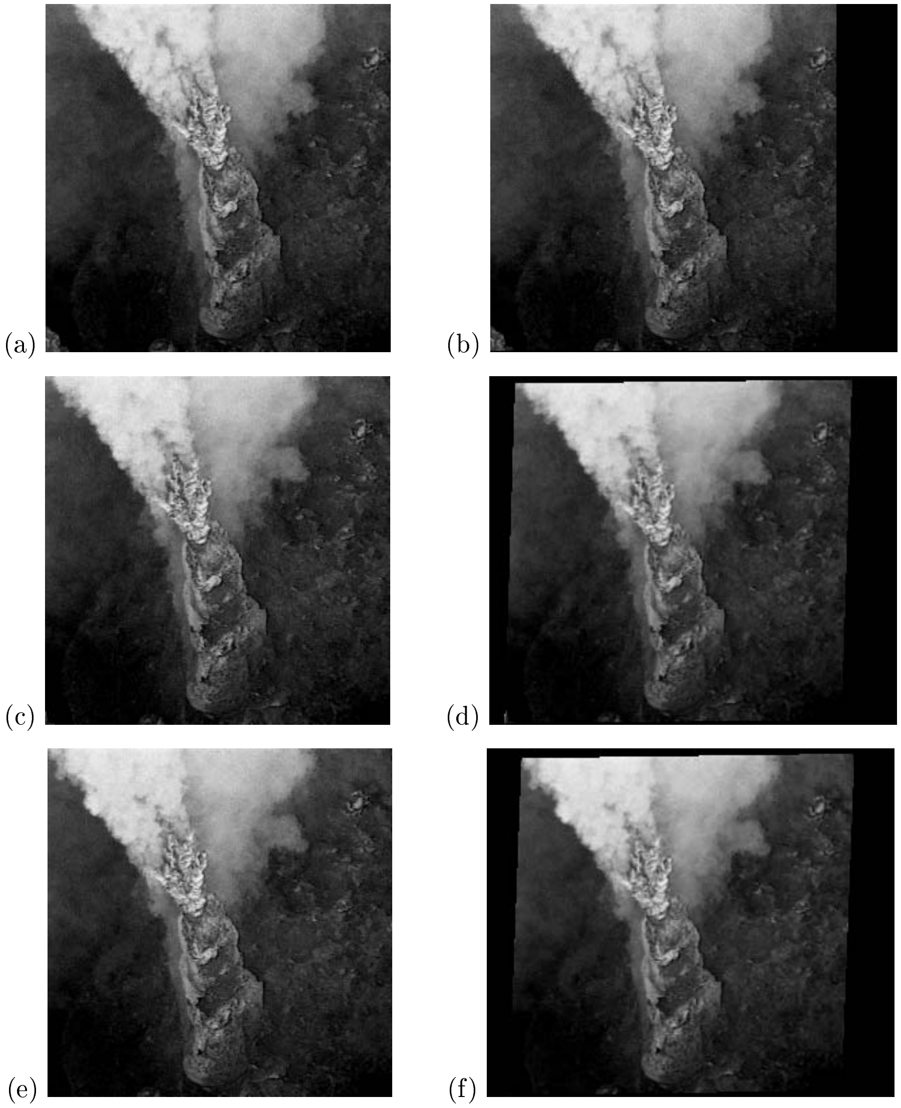


Figure 3: a) c) e) Original images of the “rock” video sequence at time t_1, t_{13} and t_{18}. b) d) f) Images compensated using the estimated dominant 2D affine motion model at time t_1, t_{13} and t_{18}.

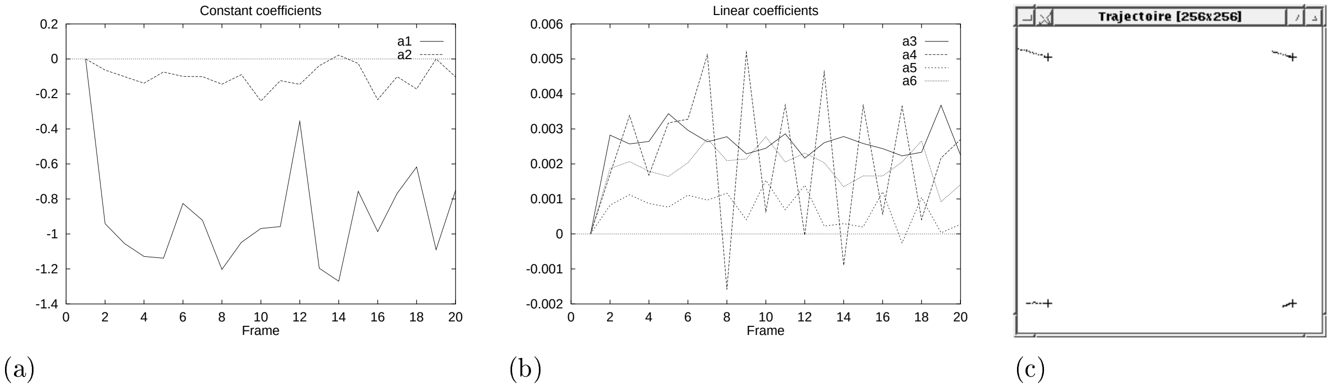


Figure 4: *“rock” sequence* ; a) Estimated constant coefficients, b) and estimated linear coefficients of the dominant 2D affine motion model. c) Apparent trajectories of points specified by the user.

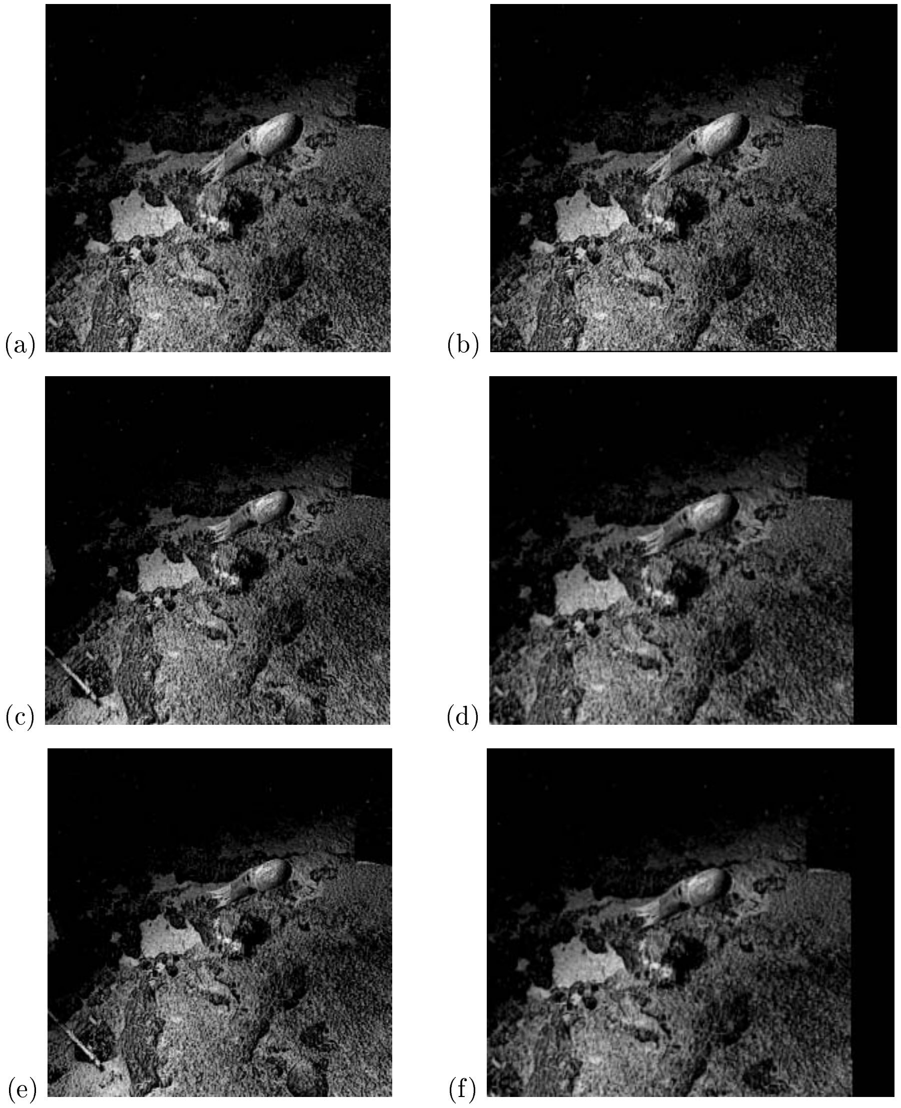


Figure 5: a) c) e) Original images of the "zoom" video sequence at time t_1, t_{13} and t_{18}. b) d) f) Images compensated using the estimated dominant 2D affine motion model at time t_1, t_{13} and t_{18}.

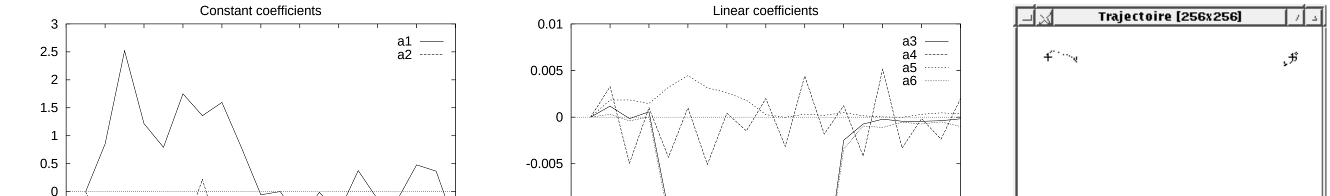


Figure 6: *"zoom" sequence* ; a) Estimated constant coefficients, b) and estimated linear coefficients of the 2D dominant affine motion model. c) Apparent trajectories of points specified by the user.