Machine Vision and Applications (2003) 14: 185–191
Digital Object Identifier (DOI) 10.1007/s00138-002-0117-7

Machine Vision and Applications

3D motion estimation of bubbles of gas in fluid glass, using an optical flow gradient technique extended to a third dimension

J. Otero, A. Otero, L. Sanchez

Computer Science Department, Oviedo University, Campus de Viesques, 33203 Gijon, Asturias, Spain; e-mail: jotero@lsi.uniovi.es

Received: 9 July 2001 / Accepted: 5 August 2002
Published online: 3 June 2003 –

Abstract. To solve the problem of estimating velocities of gas bubbles in melted glass, a method based on optical flow constraint (OFC) has been extended to the 3D case. A single camera, whose distance to the fluid is variable in time, is used to capture a sequence of frames at different depths. Since objects are not static, we cannot obtain two frames of different height values at the same time, and to our knowledge, this prevents the use of common 3D motion estimation techniques. Since the information will be rather sparse, our estimation takes several measures around a given pixel and discards the erroneous ones, using a robust estimator. Along with the exposition of the practical application, the estimation proposed here is first contrasted in the 2D case to common benchmarks and then evaluated for a synthetic problem where velocities are known.

Key words: 3D motion parameter – 3D reconstruction – Depth computation – Optical flow – robust estimation

1 Preliminary

There are basically three ways to perform the calculation of the optical flow field in the case of 2D conventional images:

- Gradient-based techniques
- Correlation-based techniques
- Frequency-based techniques

Gradient-based techniques use the well-known optical flow constraint (OFC) in order to compute the optical flow [14]. This technique makes the assumption that intensity changes in a sequence of images are due only to the movement of the objects in the scene: each pixel, corresponding to a given point of an object, will have constant brightness in the different positions that it takes during the sequence. Mathematically this is expressed as shown in Eq. (1).

$$\frac{\mathrm{d}I(x,y,t)}{\mathrm{d}t} = 0. \tag{1}$$

As it can be seen, the spatiotemporal derivatives must be estimated. This estimation is noise sensitive, so optical flow estimation has the same problem. Later in this paper, we will analyze this expression in depth.

Correlation-based techniques try to minimize a measure of similarity (or dissimilarity) between patches taken from two consecutive frames centered on a given pixel [3,5]. The displacement that maximizes (or minimizes) the selected measure divided by the interval between the acquisition of the frames is the velocity of the pixel. This approach is computationally expensive, and its complexity grows with the square of the maximum displacement searched. There are approaches that solve this problem using a bidimensional LUT (look-up table) instead of performing floating-point calculations. The gray-scale depth must be limited to maintain the LUT within a convenient size [19], but the quality of the estimation decreases as the number of gray levels does.

Frequency-based techniques use a set of tuned spatiotemporal filters to search for the velocity of a pixel [13]. Each one of the filters will give a response to the stimuli of the data (the sequence of images); the filter with the maximal response will be tuned with the velocity searched; once the filter is identified, so is the velocity. Some researchers are of the opinion that this is the most precise approach, but it is very expensive in terms of computational cost [8]. A comparison of these techniques can be found in [7].

3D optical flow estimation is not different in essence from 2D; thus, the three families of techniques could be used, but it is known that information regarding the spatiotemporal derivatives have an important role in 3D motion estimation [15].

We will show that previous approaches like [1,2,23] cannot be directly applied to the practical problem that originated this work, because of reasons that will be made clear later (Sect. 1.2).

1.1 Overview of gradient-based techniques

In 2D problems, the OFC in Eq. (1) can be expanded and written in the form

$$-\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial f}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} = \frac{\partial f}{\partial x}u + \frac{\partial f}{\partial y}v = \nabla(f)\cdot\bar{c}, \tag{2}$$

where $\nabla(f)$ is the spatial gradient of the image, "$\cdot$" is the dot vector product, $\bar{c}$ is the velocity of the pixel and $\frac{\partial f}{\partial t}$ is the temporal derivative of the sequence for that pixel.

This work has been supported by Saint Gobain Cristaleria S.A., under contract FUO-EM-034-01 with Oviedo University, Spain.

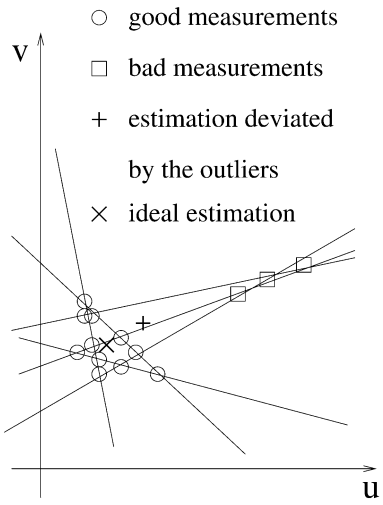

Fig. 1. Clusters of intersections of OFC pairs. Some of them belong to a dominant cluster; a few can cause deviation of the estimation if they are not rejected

The previous expression shows how the spatiotemporal derivatives are related to the observed velocities.

There are two unknowns (u and v) in this equation, so it is not possible to solve it in order to recover the full motion vector. This is known in the literature as the *aperture problem* [16], meaning that there is no way to recover the complete optical flow vector using only local (one pixel) information. With local information only the motion in the direction of the gradient (known as *normal flow*) can be computed. Many authors try to solve the aperture problem by adding some kind of global information, involving a process of regularization. That is, given a measure of some error, the process of regularization consists of applying extra restrictions to the sequence of images, searching for a minimum in the measure of that error. One of the first algorithms found in the literature using this technique is the one by Horn and Schunk, in the early 80s [14]. They apply a restriction that consists in maximizing a measure (a global smoothness criteria) and minimizing another one (the error given by the fit to OFC); these restrictions are applied all over the image.

Other researchers use an estimation of the velocities with a confidence measurement, so for each measure it is known how reliable it is [3].

There are authors that use other invariants than pixel intensity, like the zero crossings of the Laplacian of the Gaussian [11]. The motivation is that they are closer to the biological facts of animal vision systems.

Another alternative is to analyze the measures in the space of the velocities; that is, it tries to find a robust estimation of the velocity by performing an analysis on the results of many systems of OFC equations, each one applied to a pair of pixels (see Fig. 1), or it tries to fit the data to a model in order to estimate the velocity. In this way, the analysis is performed directly in the data domain that we want to recover, which is the (u, v) space [21]. Finally, some researchers perform a clustering of the OFCs themselves in order to find the most reliable one; once obtained, the corresponding normal flow to that OFC is obtained [9,18].

We will follow the first approach, using a 3D model of the bubbles and an extended OFC to a third dimension (3D OFC).

Fig. 2. Reduced-scale model of the installation where the process is analyzed

1.2 Statement of the problem and summary

This study originated in a problem in glass manufacturing. We were asked to count the number, dimension and velocities of gas bubbles in melted glass, in order to control working parameters of the oven. In this paper, we are only concerned with the third objective, measuring velocities.

The velocity of the bubbles of gas is of interest in a glass factory because the motion is related to two features of the process: the quantity of glass produced (the bubbles move with it) and the amount of heat needed to make the glass fluid enough to let the bubbles move from bottom to top and leave it. The installation used to measure all parameters is shown in Fig. 2, and its design is outlined in Fig. 3. Fluid glass falls from the upper to the lower part of the oven. There is a window from which one single camera can take images. The camera can be manually focused to any depth in the fluid, and there is room enough to install a motor that allows the camera to travel along the line shown in the schema. The velocity of the camera is 2 mm/s. In any case, we cannot take two simultaneous photographs of the fluid, and thus images will be taken in both different time and depth.

We are faced with the problem of deciding whether the addition of the automatic displacement would allow measuring velocities with a reasonable precision. To evaluate this possibility, we have synthetized a set of frames that simulate the view of the melted glass from the motorized camera and designed an algorithm able to process them. In Fig. 4, an image obtained in the real installation is juxtaposed with a synthetic frame. Bubbles are at a different z, and because of this, they are blurred with a different intensity.

Intuitively, it is clear that the precision will not be as high as it could be in the case where we could install two cameras, but it can still be of practical interest. It is remarked that existing 3D reconstruction techniques, like optical sectioning usually found in microscopy (see [10]), cannot be applied, and the same can be said about stereo-based approaches [2,23], as we anticipated in the introduction. Approaches like [1] cannot be applied to our problem because the motion of the camera in that work is fundamental to the behavior of the algorithm.

In the next sections, the 3D reconstruction method is explained, and after it the velocity computation is presented. The algorithm will be applied to solve a synthetic problem for

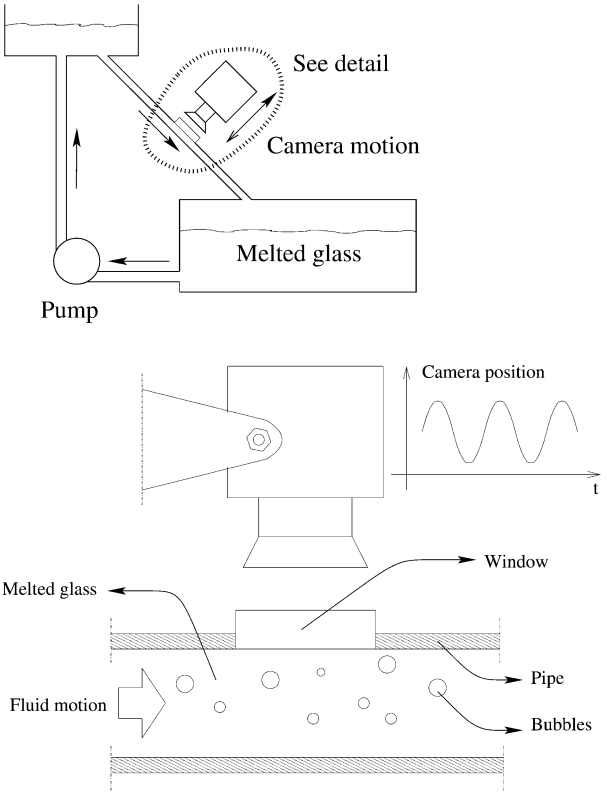

Fig. 3. *Top*: Schema of glass circulation in the model of the installation. The camera is mounted perpendicular to the pipe. *Bottom*: Detail of elements of interest for the problem

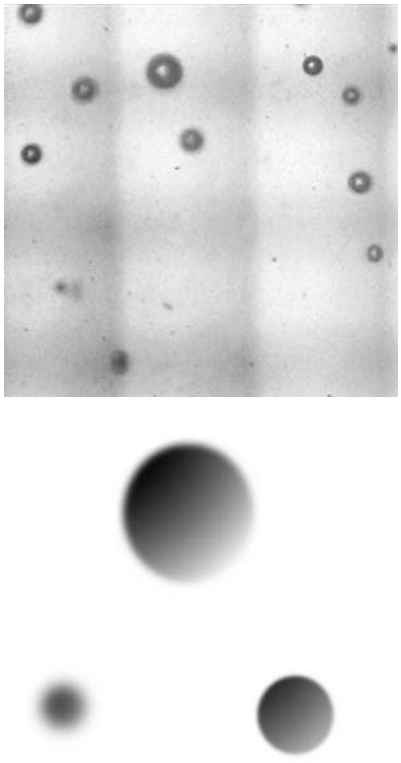

Fig. 4. *Top*: Bubbles of gas in melted glass at different depth. Note the different degree of blurriness. *Bottom*: Synthetic frame

which the true velocity field is known in order to estimate its precision, and then it is applied to the real data. Additionally, some of the decisions adopted when estimating OFC intersections will be contrasted to their equivalents in different optical flow algorithms for standard 2D benchmarks [7].

2 3D motion estimation from 3D reconstruction and 3D OFC

In this section, we explain how to obtain 3D velocity from 3D data, and how to reconstruct 3D data from 2D sections taken at different times.

2.1 3D extension of OFC

In order to perform our 3D motion analysis, the same assumption relied upon for the usual OFC is extended to the 3D model, and thus we postulate that the intensity of the bubbles reconstructed is constant over the time. In this way, the sequence becomes a function in x, y, z, t and the 3D OFC is now the expression found in Eq. (3). In this equation, f is the intensity of the pixels in the 3D model and $\overline{c}_{3D}$ is the 3D velocity of the pixels.

$$-\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial f}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} + \frac{\partial f}{\partial z}\frac{\mathrm{d}z}{\mathrm{d}t} = \frac{\partial f}{\partial x}u + \frac{\partial f}{\partial y}v + \frac{\partial f}{\partial z}w = \nabla(f)\cdot\overline{c}_{3D}. \quad (3)$$

As in the case of the usual OFC [see Eq. (2)], there are too many unknowns in the 3D OFC to solve it and obtain c_{3D}. The same techniques that try to solve the OFC for u and v can be extended to obtain u, v and w from 3D OFC.

2.2 3D reconstruction from sections taken at different time and depth

The 3D intensity reconstruction is estimated from several images taken with a different z, which is with a different focal distance and with a narrow field depth. In this way, we get a 2D projection of the objects at each z, where a given focal setting gives the highest precision in the image acquisition. In Fig. 5, this process is schematized. The volume of fluid being studied is represented by a box with a single bubble. The left side of the figure, shows the situation at time t_i, and the right side of the image shows the position of the bubble at time t_{i+1}. The focus moves down and the bubble moves to the left top corner. In each situation, the position is shown where the focus brings finer detail. The resultant 2D image is shown above each volume of fluid.

Since the different images are not taken at the same time, we cannot perform a numerical estimation of the spatiotemporal derivatives using a domain around each pixel and using the difference between values in convenient directions as derivatives. Instead of this, we compute an estimation of the intensity at each (x, y, z) from images taken at different t, assuming a

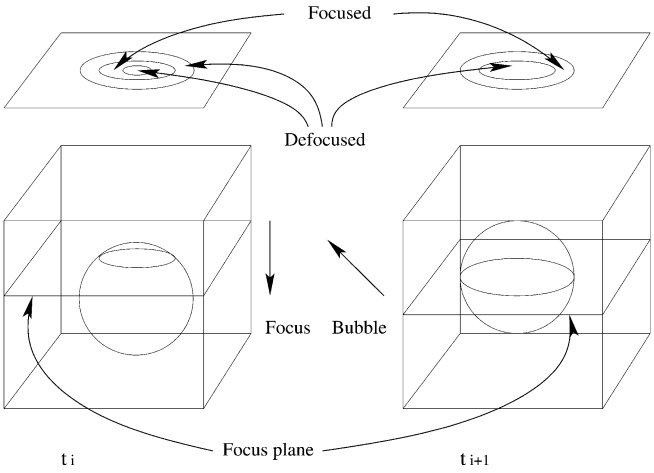

Fig. 5. Acquisition of the images at different z. The relative motion between the focus plane and the bubbles bring different degrees of blurriness

lineal evolution of the intensity as can be seen in Eq. (4).

$$f(x, y, z, t) = k_1 x + k_2 y + k_3 z + k_4 t + k_5. \qquad (4)$$

It should be clear that, after the model has been estimated, k_i is the intensity derivative (i varying from 1 to 4) as can be seen in Eq. (5).

$$\frac{\partial f}{\partial x} = k_1; \quad \frac{\partial f}{\partial y} = k_2; \quad \frac{\partial f}{\partial z} = k_3; \quad \frac{\partial f}{\partial t} = k_4. \qquad (5)$$

As a minimum, we need a set of five equations like Eq. (4) to solve the linear equation system for the unknowns k_i. This leads us to the equations in Eq. (6).

$$\begin{array}{lcl} f(x_1, y_1, z_1, t_1) & = & k_1 x_1 + k_2 y_1 + k_3 z_1 + k_4 t_1 + k_5; \\ f(x_2, y_2, z_2, t_2) & = & k_1 x_2 + k_2 y_2 + k_3 z_2 + k_4 t_2 + k_5; \\ f(x_3, y_3, z_3, t_3) & = & k_1 x_3 + k_2 y_3 + k_3 z_3 + k_4 t_3 + k_5; \\ f(x_4, y_4, z_4, t_4) & = & k_1 x_4 + k_2 y_4 + k_3 z_4 + k_4 t_4 + k_5; \\ f(x_5, y_5, z_5, t_5) & = & k_1 x_5 + k_2 y_5 + k_3 z_5 + k_4 t_5 + k_5. \end{array} \qquad (6)$$

In order to improve the quality of the results, we choose to perform an estimation over a larger set of data, using a neighborhood of pixels that hold enough information in each direction of the parameter space. Because the resolution of the data is greater in the x and y directions than in z and t directions, we take a greater amount of data in these directions, in order to gain enough information about variations of intensity through z and t. In our experiments, we have found that a neighborhood like the one in Fig. 6 is suitable for the estimation of k_i. In this neighborhood, we take all the data in z and t directions that have x and y coordinates from the set $(i, j), (i+1, j), (i, j+1)$. For each velocity estimation, the camera takes images at 10 different z's, so we have a total of 30 equations. This is an over-constrained set of equations that can be solved to find k_i, the spatiotemporal derivatives that we need to use in Eq. (3). The final over-constrained system is the one in Eq. (7).

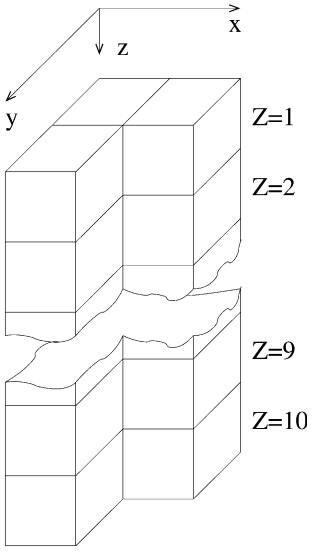

Fig. 6. Neighborhood for k_i estimation of the bubbles model from a set of images at different depth

$$\begin{array}{lcl} f(j, i, 1, 1) & = & k_1 j + k_2 i + k_3 + k_4 + k_5 \\ f(j, i, 2, 2) & = & k_1 j + k_2 i + 2k_3 + 2k_4 + k_5 \\ \ldots & & \\ f(j, i, 10, 10) & = & k_1 j + k_2 i + 10k_3 + 10k_4 + k_5 \\ f(j, i+1, 1, 1) & = & k_1 j + k_2 (i+1) + k_3 + k_4 + k_5 \\ f(j, i+1, 2, 2) & = & k_1 j + k_2 (i+1) + 2k_3 + 2k_4 + k_5 \\ \ldots & & \\ f(j, i+1, 10, 10) & = & k_1 j + k_2 (i+1) + 10k_3 + 10k_4 + k_5 \\ f(j+1, i, 1, 1) & = & k_1 (j+1) + k_2 i + k_3 + k_4 + k_5 \\ f(j+1, i, 2, 2) & = & k_1 (j+1) + k_2 i + 2k_3 + 2k_4 + k_5 \\ \ldots & & \\ f(j+1, i, 10, 10) & = & k_1 (j+1) + k_2 i + 10k_3 + 10k_4 + k_5 \end{array} \qquad (7)$$

Once having calculated the spatiotemporal derivatives k_i, we can estimate the 3D velocity using Eq. (3).

3 Robust estimation of optical flow

As we stated before, the nature of the available data in our problem complicates the computation of the optical flow. This is due to the sparsity of the sections, which cannot bring us a perfect reconstruction of the 3D structure of the bubbles. Since the parameters that we recover from the different sections, k_i, are the spatiotemporal derivatives that appear in Sect. 2.1, the resulting optical flow can be affected by the errors during the 3D reconstruction. Because of this, we have to develop an algorithm that discards the erroneous measurements that may occur. We decided to analyze the density of the distribution of velocities and to search for its maximum (the mode of the distribution) instead of calculating its mean value, which is faster to obtain but is rather influenced by outliers, which in the problem addressed here are frequent.

In our problem, another circumstance arises: the fact that there is several objects (the bubbles) moving over an static background. In this case, it is not possible to use approaches like [14] because the global smoothness criteria does not hold in those boundaries. Authors, like Nagel, propose to relax the

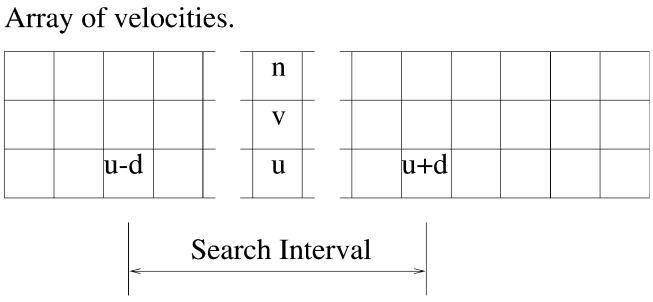

Fig. 7. This is the array where the velocities are sorted in order to improve the performance of mode calculation. Because of the sorting criteria, we only need to examine the velocities that belong to the search interval

smoothness criteria in those places where the gradient is high, smoothing the flow along the contour but not through the contour [17]. But the problem remains: due to the fact that the OFC has two unknowns, it's necessary to take measurements from at least two pixels. If the pixels are chosen from different objects with different velocities (for instance, a bubble and the background), the solution to the system of equations obtained will give an erroneous velocity. The analysis of the velocity distributions present within a neighborhood of each pixel (in the direction addressed in [21]) is necessary to determine which velocities are supposed to be correct and which one is the dominant velocity in the neighborhood (see Sect. 3.1). Most of the previous work in this area, and all standard benchmarks, are based in 2D data. To assess the effectiveness of the mode as a robust estimator, we will first reformulate our algorithm for the 2D case and then compare it to other algorithms, using images found in [4] and [7].

3.1 Mode as an estimator of optical flow

The maximum of the density function of the velocity field can be estimated by finding the higher cell of the bidimensional (in 2D) histogram of the velocity field, but this estimation can be made both faster and more precise if we numerically maximize either a k-neighbor or a kernel estimation of this density [22]. We tried both of them, with a compact support kernel in the latter case:

$$k(\overline{x}) = \begin{cases} \frac{|\overline{x}_0 - \overline{x}|}{d} \cdot y(\overline{x}) & \text{if } |\overline{x}_0 - \overline{x}| < d \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

As in the previous method, we compute the solutions of every possible pair of OFCs (except the systems whose condition number is to high) for a given neighborhood. We discard the solutions whose module is greater than a threshold as errors. The solutions to the system of OFCs solved are stored in an array with three columns. In the first column, we store the velocity component in the x direction, and in the second column, we store the velocity component in the y direction as shown in Fig. 7 (the array is rotated 90 degrees in the figure.)

Using compact support kernels allows us to improve the efficiency of the algorithm, because we can limit the number of evaluations of Eq. (8) needed to find the mode. Since the kernel is 0 for points further than d, we only need apply the kernel to the velocities closer than this value. If the array is sorted by the first component (u) and then by the second (v) (the third field is used to store the sum of the kernel function centered at that velocity for all points in the array), we can restrict ourselves to velocities for which the difference in the first component is smaller than d (see Fig. 7). The mode will be the pair (u, v) for which the value of the third field is maximum. The value of the maximal density is also a measure of the reliability of the estimation: the bigger the value of the maximal is, the more reliable the estimation is. This can be seen in Fig. 8, where the density of neighbors obtained (in x axis) versus the error of an experiment (in y axis) is shown. Clearly, as the value of the density grows, the error decreases. The test sequence used was "MysineC-16," taken from the ones used in [7]. In this sequence a bidimensional sinusoidal pattern moves at $(1, 1)$ pixels per frame.

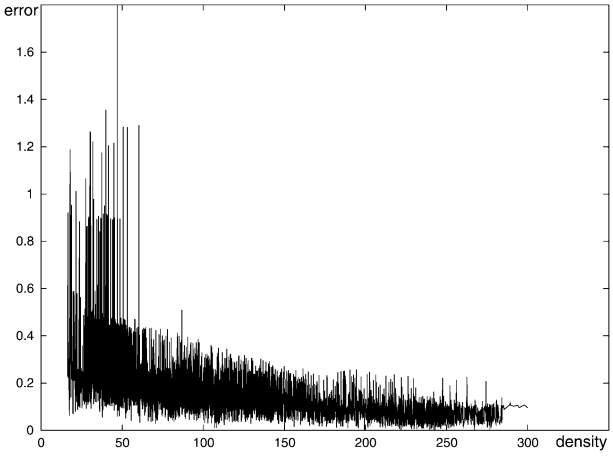

Fig. 8. Plot of the density value versus the error of the estimation. As can be seen, there is a decreasing tendency in the graphic

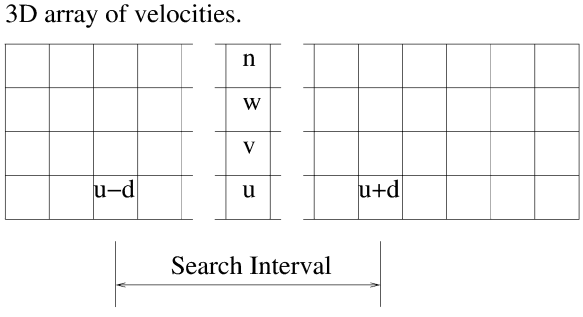

Fig. 9. Array where the 3D velocities are stored and sorted in order to improve the efficiency of the algorithm

Observe that the 3D formulation is immediate. There are three unknowns in the 3D OFC, therefore we need at least three equations to compute the velocity of a point of the reconstructed 3D solid. In Sect. 3.1, we computed all the pairs of OFC that can be found in a 2D neighborhood of a pixel. Now we will compute all the possible triplets of equations that can be found in a 3D neighborhood of a point of the reconstructed solid. Then, for each point of the reconstructed solid, we have a 3D distribution of velocities, whose mode must be determined. Velocities are stored now in an array with four rows that are sorted using a similar criteria as in the 2D algorithm (see Fig. 9). After the sorting stage, a kernel like the one in Eq. (8) is applied to the 3D data ($\overline{x}$ and $\overline{x}_0$ will be 3D vectors in this case). Only the data belonging to the search

interval $(u - d, u + d)$ has to be used to compute the density estimation of each velocity, exactly as before.

4 Results

The 2D version of the algorithm is compared to other algorithms first, and the 3D version will be used to solve a synthetic problem later. The objective of the 2D comparison is to determine whether the mode is indeed a more robust estimator than other approaches. The synthetic problem is introduced to determine if the estimation of the velocities of the bubbles is precise enough for the practical application.

4.1 Comparative results, 2D algorithm

In order to perform a quantitative comparison with other algorithms, we adopt the metric defined in [20]; other measures (for example, [7]) would serve as well. In short, the error over an image is the sum of errors of all pixel, and the error on a pixel is the Euclidean distance between estimated and real velocities.

The test sequences used in these series of test where "Translating tree," "Diverging tree," "Yosemite flight thru," "Rotating sphere," "Diverging office," and "Street." The first three are used in [7]. The sequences are semi-synthetic, real images manipulated to give the illusion of movement in the scene or by the camera. The last three ones were proposed in [4] as an alternative to perform the same task. We chose these sequences to perform the same analysis published in [7]. The source code used in the test of Anandan's algorithm is the same as used in [7]. To test the algorithms in [21] and [18], we used our own implementation. In Table 1, numerical values of the error are compared to those algorithms from [3,18, 21]. The neighborhood size was 5×5 for both Anandan's algorithm and our approach. For the other approaches the size of the neighborhood was 15×15. The size of the neighborhoods was chosen in order to provide the algorithms with a sufficient amount of data (comparable to the quantity used in our approach) to perform the estimation as proposed in [18] and [21]. The estimator presented here performs better than Anandan's algorithm in the case of the "Translating tree," "Diverging tree" and "Diverging office" sequences. In the case of "Rotating sphere" our approach performs better than OFC's parameter clustering. In the case of "Yosemite flight through" and "Street," Anandan's algorithm performs slightly better. The k-neighbor approach is slightly worse than the kernel estimator, but it is also more computationally efficient.

4.2 Quantitative results, 3D synthetic data

It was mentioned that, in order to know the attainable precision when estimating 3D motion with one only camera, we had to build a synthetic problem for which the true motion is known. This lets us test our algorithm prior to the full implementation of the system in the future.

In Fig. 10, three synthetic images are shown. The top and center parts of the image are simulated photographs of a single bubble at different heights, moving from the lower left corner to the upper right corner and toward the observer; the velocity of all the points of the image is $(1, 1, 1)$. The bottom part of the figure is reconstructed from the sequence of images from which the two previous ones were taken, assuming that the hypotheses in Eq. (4) concerning linear change in the movement are true. The bubble in that figure moves with velocity $(1, 1, 1)$.

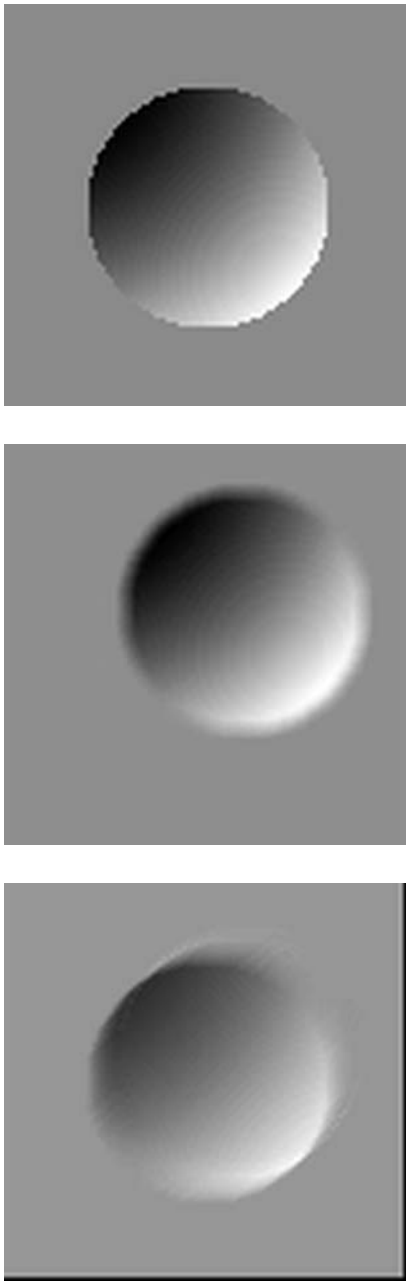

Fig. 10. Example of image reconstruction from the synthetic 2D data. An image at $z = 5$, $t = 1$ is reconstructed with the model in Eq. (4) from the whole set of 2D images

In Fig. 11, the error histogram for the component in the z direction obtained from the reconstructed 3D data is shown. The total error in the image was 0.15×10^4, that means a percent error of 15% for an image size of 100×100.

The obtained error is good enough for the purpose in the factory process and brings important information regarding the amount of heat needed by the process. Real precision will

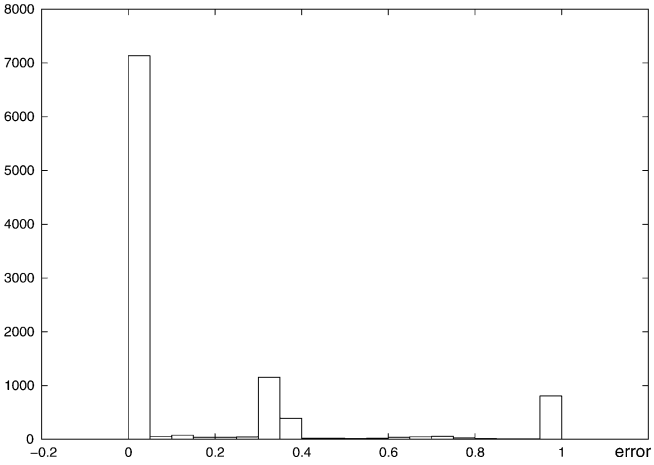

Fig. 11. Histogram of the error in the optical flow computation from sequence in Fig. 10. Note the peaks due to imprecision of the model

Table 1. The error obtained (divided by 10^4) in the test by kernel estimator and variable kernel, compared to Anandan's algorithm [3], OFC's slope intercept parameter clustering [18] and intersections over central pixel OFC clustering [21]

Sequence	Schunck	OFC par. clustering	Anandan	Kernel	Var kernel
Trans. tree	6.7663	4.4438	3.9809	2.3039	2.4032
Div. tree	3.5310	2.8188	1.5067	1.4246	1.5195
Yosemite	16.090	15.580	10.154	10.985	11.287
Rot. Sph.	1.8011	1.0137	2.4696	0.6612	0.7408
Office	5.1679	3.8374	2.1625	1.9104	2.1828
Street	5.8890	4.7729	1.8539	2.1046	2.4034

be lower, since noise was not present in synthetic data (see Fig. 4), but these results suggest us that the method is useful for practical purposes.

5 Concluding remarks

Due to the particular conditions of the process discussed here (one single camera, absence of references), it was not possible to apply common 3D motion estimation procedures to determine the velocities of bubbles in melted glass. Since the camera can focus at different depths in the fluid but cannot take two simultaneous photographs of the fluid, our algorithm had to cope with sequences of images taken at both different times and depth. This restriction was solved when some linearity constraints were assumed and a method for estimating all needed partial derivatives comprising the OFCs was developed. This provided us with the data needed to state and solve a problem formulated in terms of OFC equations with three unknowns. The result is a new algorithm for estimating 3D optical flow, based on robust gradient techniques. The precision of this method, while being inherently inferior to that of stereo procedures, was high enough for it to be applied in practice.

References

1. Adiv G (1989) Determining three-dimensional motion structure from optic flow generated by several moving objects. IEEE Trans Pattern Anal Mach Intell 11(5):477–489
2. Ahuja N, Abbott AL (1993) Active stereo: integrating disparity, vergence, focus, aperture, and calibration for surface estimation. IEEE Trans Pattern Anal Mach Intell 15(10):1007–1029
3. P Anandan (1985) A computational framework an an algorithm for the measurement of visual motion. Int J Comput Vision 2:283–310
4. McCane B, Galvin B, Novins K (1998) On the evaluation of optical flow algorithms. Computer Science Department, University of Otago, New Zealand
5. Ancona N, Poggio T (1993) Optical flow from 1D correlation: application to a simple time-to-crash detector. Massachusetts Institute of Technology, Artificial Intelligence Laboratory and Center for Biological and Computational Learning
6. Battiti R, Amaldi E, Koch C (1991) Computing optical flow across multiple scales: an adaptive coarse-to-fine strategy. Int J Comput Vision 6(2):133–145
7. Barron JL, Fleet DJ, Beauchemin SS (1995) Performance of optical flow techniques. Int J Comput Vision 1(12):43–77
8. Beauchemin SS, Barron JL (1996) The computation of optical flow. ACM Comput Surv 3(27):433–467
9. Ben-Tzvi D, Del Bimbo A, Nesi P (1993) Optical flow from constraint lines parametrization. Pattern Recogn 26(10):1549–1561
10. Weinstein M, Castleman KR (1971) Reconstructing 3D specimens from 2D section images. Proc SPIE 26:131–138
11. Duncan JH, Chou T-C (1992) On the detection of motion and the computation of optical flow. IEEE Trans Pattern Anal Mach Intell March 14(3):346–352
12. Enkelmann W (1986) Investigations of multigrid algorithms for the estimation of optical flow fields in image sequences. In: Proceedings of workshop on motion, representation and analysis, Charleston, S.C., 7–9 May 1986. IEEE Computer Society Press, Los Angeles, Calif.
13. Heeger DJ (1988) Optical flow using spatiotemporal filters. Int J Comput Vision 1:279–302
14. Berthold K P Horn, Schunk BG (1980) Determining optical flow. A.I. Memo 572. Massachusetts Institute of Technology, Artificial Intelligence Laboratory
15. Koenderink JJ, van Doorn AJ (1987) Facts on optic flow. Biological Cybern 56:247–254
16. Murray DW, Buxton BF (1990) Experiments in the machine interpretation of visual motion. MIT Press, Cambridge, Mass.
17. Nagel HH (1983) Displacement vectors derived from second-order intensity variations. Comput Vision Graph Image Process 21(1):85–117
18. Nesi P (1995) Real-time motion estimation. Department of Systems and Informatics, Faculty of Engineering, University of Florence
19. Otero J, Cancelas JA, Gonzalez RC (1998) Optical flow calculation using look-up tables. In: Ollero A (ed) IFAC workshop on intelligent components for vehicles, Seville, Spain, 23–24 March 1998. Pergamon, New York
20. Otte M, Nagel HH (1994) Optical flow estimation: advances and comparisons. In: Eklundh J-O (ed) European conference on computer vision, Stockholm, Sweden, 2–6 May 1994. Springer, Berlin Heidelberg New York
21. Shunck BG (1989) Image flow segmentation and estimation by constraint line and clustering. IEEE Trans Pattern Anal Mach Intell 11(10):1010–1027
22. Silverman BW (1986) Density estimation for statistics and data analysis. Chapman and Hall, London
23. Wang W, Duncan JH (1996) Recovering the three-dimensional motion and structure of multiple moving objects from binocular image flows. Comput Vision Image Understanding 63(3):430–446