

# Real-time parallel computation of disparity and optical flow using phase difference

F. Valentinotti, G. Di Caro, B. Crespi

Institute for Scientific Research and Technology (IRST), I-38050 Povo, Trento, Italy

**Abstract.** A phase-difference-based algorithm for disparity and optical flow estimation is implemented on a TI-C40-based parallel DSP system. The module performs real-time computation of disparity maps on images of size  $128 \times 128$  pixels and computation of optical flows on images of size  $64 \times 64$  pixels. This paper describes the algorithm and its parallel implementation. Processing times required for the computation of disparity maps and velocity fields and measures of the algorithm's performance are reported in detail.

**Key words:** Phase-based image matching – Disparity – Optical flow – Parallel processing

## 1 Introduction

The capability to perceive depth and movement is a necessary requirement for an autonomous system moving in a real-world environment [9]. Binocular vision is one of the basic mechanisms for depth perception. The distance of an object can be estimated from the position disparity between its images projected on the left and the right visual sensors. Analogously, the velocity of an object moving in the visual field can be evaluated from the computation of the displacement of the object's image between successive time intervals.

The evaluation of disparity maps and optical flows are computationally intensive operations because they involve the search for corresponding points in pairs of images taken from different points of view, displaced in space for disparity or in time for the optical flow. Since the basic computation is the same, the same computational technique can be used to treat both problems. In this way, software and hardware requirements for the implementation of the procedures can be better optimized and partial results can be shared by the different processing modules.

The phase-difference-based method [6, 14] is a suitable and attractive choice for image-matching applications. It produces dense disparity and velocity maps. It has a broad range of applicability because it does not depend on the choice of

visual features specific to classes of images. It is robust under contrast variations and small-scale changes between the image pair. Furthermore, internal parameters can be easily adjusted to optimize the system response to changing inputs.

In this work, an algorithm based on the phase-difference technique introduced in [6, 7] is developed and implemented on a parallel network of TI-C40 [16]. The TI-C40 is a powerful digital signal processor (DSP) with six fast communication ports. Due to its flexibility, modularity and computational power, the C40 module is particularly suitable for real-time image processing in embedded systems.

The parallel system, composed of five processors plus a frame grabber, performs real-time disparity estimation on images of  $128 \times 128$  pixels and real-time computation of the optical flow on images of dimension  $64 \times 64$ . Here, real time means that the *reaction time* of the system to a visual stimulus is comparable to the reaction time of a human being, which is approximately 100–200 ms.

The phase-based technique, in one and two dimensions, is described in Sects. 2 and 3, respectively. The hardware configuration is presented in Sect. 4. The algorithm's implementation, processing times, and system performance are reported in Sect. 5 for the estimation of the disparity maps and in Sect. 6 for the estimation of velocity fields.

## 2 Phase-difference-based stereopsis

In the overlapping region of their visual fields, two stereo images show the same visual patterns seen from slightly different perspectives. The positions of corresponding points on the left and on the right images are related by a one-dimensional shift, the position disparity, along the direction of the epipolar lines. If the optical axes of the cameras are parallel, the epipolar lines are parallel to the line joining the optical centers of the cameras.

In the phase-difference method, disparity is computed from the phase difference between the convolutions of the two stereo images with Gabor filters. The assumption that the two signals,  $I_1(x)$  and  $I_2(x)$ , are *locally* related by a shift means that in a neighborhood  $U(x_0)$  of each point  $x_0$ , the equation

$$I_1(x + \delta(x_0)/2) = I_2(x - \delta(x_0)/2) \quad x \in U(x_0) \quad (1)$$

holds. In this region, the  $k$  Fourier components of  $I_1(x)$  and  $I_2(x)$  are related by a phase difference equal to  $\Delta\phi(k) = \phi_2(k) - \phi_1(k) = k\delta$ .

A local Fourier analysis can be performed by convolving the images with Gabor filters. A Gabor filter [8] is a Fourier kernel multiplied with a Gaussian envelope,

$$G(x - x_0; \sigma, k_0) = e^{i k_0 (x - x_0)} e^{-\frac{(x - x_0)^2}{2\sigma^2}}. \quad (2)$$

The Fourier transform of  $G(x)$  is

$$\hat{G}(k - k_0; \sigma, k_0) = e^{-ix(k - k_0)} e^{-\frac{\sigma^2 (k - k_0)^2}{2}}. \quad (3)$$

The Gabor filter is localized at  $x = x_0$  in space and at  $k = k_0$  in frequency space. There are two free parameters:  $\sigma$  represents the half-width of the Gaussian envelope and  $\nu_0 = k_0/2\pi$  is the spatial frequency at which the filter is tuned, i.e. the peak frequency.

The bandwidth in the frequency space is given by  $\tau = 1/\sigma$ . A relation between  $\sigma$  and  $k_0$  is imposed by assuming that the bandwidth of the Gabor filter is one octave, that is

$$\tau = \frac{1}{\sigma} = k_0 \frac{2^\beta - 1}{2^\beta + 1}, \quad \text{with } \beta = 1.$$

This means the Gaussian envelope contains a complete wavelength  $\lambda$  in the interval  $[x_0 - \sigma, x_0 + \sigma]$ ,  $\lambda = \frac{2\pi}{\tau}\sigma \approx 2.1\sigma$ .

The Gabor filter is the local bandpass filter that is best localized in both space and frequency. For this filter, the product of the uncertainties in space and in frequency attains the minimum value,  $\Delta x \cdot \Delta k = 1$ . Non-Gaussian envelopes may have better spatial resolution or they may have better resolution in the frequency domain, but the Gaussian envelope gives the best overall result.

The results of the one-dimensional convolution along the lines of the images,

$$C_i(x; k_0) = \int G(x - z; k_0) I_i(z) dz = \rho_i(x) e^{i\psi_i(x)} \quad (4)$$

$i = 1, 2,$

are complex functions characterized by amplitudes  $\rho_i(x)$  and phases  $\psi_i(x)$ , index  $i$  indicates the left and right components. For notation simplicity, the index will be dropped in the following when we refer to quantities with left and right components. Phases, being defined by the ratios between the real and imaginary part of the convolution  $C$ ,

$$\psi(x) = \arctan \left[ \frac{\text{Im } C(x)}{\text{Re } C(x)} \right] \quad (5)$$

are stable under smooth changes of intensity.

As a function of the spatial position, the phase of the filter response,  $\psi(x)$ , has a quasi-linear behavior dictated by the peak frequency  $k_0$ ,

$$\psi(x) \approx \psi'(x_0)(x - x_0) \approx k_0(x - x_0). \quad (6)$$

In fact, the local frequency, i.e. the derivative of the phase  $\psi(x)$ , is generally close to the value of the peak frequency  $k_0$ .

Disparity is extracted from the phase difference,  $\Delta\psi(x) = \psi_2(x) - \psi_1(x)$ , by expanding  $\Delta\psi(x)$  to the second order in  $\delta$ ,

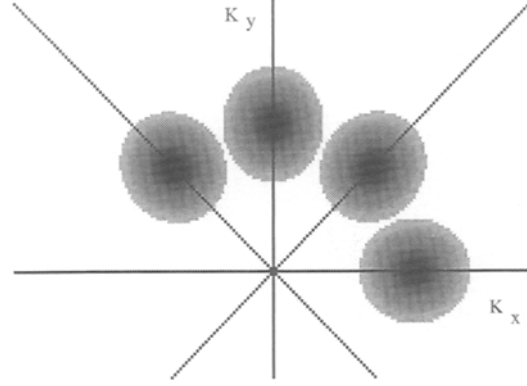


Fig. 1. Gray-level image for a set of  $N_F = 4$  directional Gabor filters. Only values in the range  $[0.5, 1]$  are shown

$$\delta(x) = \frac{[\Delta\psi(x)]_{2\pi}}{\psi'(x)}. \quad (7)$$

The phase is not defined when the amplitude vanishes, i.e. when  $\rho^2(x) = (\text{Im } C)^2 + (\text{Re } C)^2 = 0$ . Around these singular points the phase is very sensitive to spatial or scale variations. As a consequence, approximation (7) fails and the calculation of disparity in the neighborhood of a singularity is unreliable. The neighborhoods of singular points can be detected [6] by means of the quantity

$$S(x) = \frac{1}{\tau} \left| \frac{d}{dx} \log(C(x)) \right| = \frac{1}{\tau} \left| \frac{\rho'(x)}{\rho(x)} + i(\psi'(x) - k_0) \right|. \quad (8)$$

The imaginary part in (8) measures the difference between the peak frequency and the local frequency,  $\psi'(x)$ , in relation to the width of the filter  $\tau$ . The real part measures local amplitude variations with respect to the spatial width  $\sigma$ . Disparity calculation at point  $x$  is accepted only if this quantity is not greater than a confidence threshold,  $S(x) < T_S \approx 1.2$ .

Singularities are isolated points in the scale space  $(x, \lambda)$ . However, in regions of the image that do not contain enough information, the amplitude  $\rho(x)$  of the Gabor filters is close to zero. Since the signal-to-noise ratio is low, spatial derivatives assume large values and the quantity  $S(x)$  exceeds the threshold over the entire region. To prevent occasional errors, a constraint on the filter energy  $\rho(x)$  is also added,  $\rho(x) > T_E$ , where  $T_E$  is a threshold value.

### 3 The 2D model: optical flow estimation

The one-dimensional technique can be extended to two or more dimensions. Two-dimensional shifts (2D) occur in the computation of 2D disparities and in the computation of the optical flow, in which 2D displacements between corresponding points of images taken at successive times,  $I_1 = I(t_1)$  and  $I_2 = I(t_2)$ , have to be computed. In [3, 7], Gabor filters are extended to the temporal dimension, and a stream of images is needed. However, the 3D technique requires an excessively high computational power for real-time applications.

The two images are convoluted with a set of  $N_F$  2D Gabor filters characterized by different directions, as shown in Fig. 1. Following [7, 11], the entire set of directional Gabor filters is generated by rotating one of them around the origin, for example, the one centered on the  $K_x$  axis. The filters can be expressed by the formula

$$\begin{aligned} \hat{G}_j(\mathbf{k}; \mathbf{k}_0^j, \mathbf{x}) \\ = \exp \left[ -\frac{1}{2}(\mathbf{k} - \mathbf{k}_0^j)^T \mathbf{A}_j (\mathbf{k} - \mathbf{k}_0^j) \right] e^{i(\mathbf{k} - \mathbf{k}_0^j) \cdot \mathbf{x}}, \end{aligned} \quad (9)$$

where  $\mathbf{k}_0^j = k_0 (\cos \theta_j, \sin \theta_j)$ , and  $\theta_j = \pi(j-1)/N_F$ , with  $j = 1, \dots, N_F$ . Matrix  $\mathbf{A}_j$  is the  $2 \times 2$  matrix obtained by rotating the diagonal matrix  $\mathbf{D}$ , whose elements are the inverse of the square of the standard deviations along the radial and angular directions, by means of rotation matrices  $\mathbf{R}_j$

$$\begin{aligned} \mathbf{A}_j &= \mathbf{R}_j \mathbf{D} \mathbf{R}_j^T \\ &= \begin{pmatrix} \cos \theta_j & -\sin \theta_j \\ \sin \theta_j & \cos \theta_j \end{pmatrix} \begin{pmatrix} \frac{1}{\tau_r^2} & 0 \\ 0 & \frac{1}{\tau_\theta^2} \end{pmatrix} \begin{pmatrix} \cos \theta_j & \sin \theta_j \\ -\sin \theta_j & \cos \theta_j \end{pmatrix}. \end{aligned} \quad (10)$$

Each filter  $\hat{G}_j(\mathbf{x})$  has a preferential direction  $(\cos \theta_j, \sin \theta_j)$  and a peak frequency  $\mathbf{k}_0^j$ . The width along the radial direction is  $\tau_r = 1/\sigma_r$  and the width along the angular direction is  $\tau_\theta = 1/\sigma_\theta$ . The relation between the width along the filter's direction,  $\tau_r$ , and the peak frequency,  $k_0$ , is set by the octave number  $\beta$ ,  $\tau_r = k_0 \frac{2^\beta - 1}{2^{\beta+1}}$ . The angular bandwidth depends on the number of filters.

The convolution results are 2D complex functions,

$$\begin{aligned} C_1^j(x, y) &= G_j * I_1 = \rho_1^j e^{i\psi_1^j} \\ C_2^j(x, y) &= G_j * I_2 = \rho_2^j e^{i\psi_2^j}. \end{aligned} \quad (11)$$

A filter is *non-singular* if the local phase  $\nabla \psi^j(\mathbf{x})$  is close to the peak frequency  $\mathbf{k}_0$  and if the amplitude does not change too much in an area of size  $\sigma_r \sigma_\theta$  in the  $\mathbf{x}$  space. The definition of singular neighborhoods is the 2D generalization of (8),

$$\begin{aligned} S^j(\mathbf{x}) &= \left| (\nabla \log C^j(\mathbf{x}))^T \mathbf{A} \nabla \log C^j(\mathbf{x}) \right| \\ &= \left| \frac{\nabla \rho^j(\mathbf{x})^T \mathbf{A} \nabla \rho^j(\mathbf{x})}{\rho^j(\mathbf{x})^2} \right. \\ &\quad \left. + i (\nabla \psi^j(\mathbf{x}) - \mathbf{k}_0^j)^T \mathbf{A} (\nabla \psi^j(\mathbf{x}) - \mathbf{k}_0^j) \right| \\ &< 2 T_S \approx 2.4 \end{aligned} \quad (12)$$

where matrix  $\mathbf{A}$  is defined by (10).

Expanding to the second order, the relation between the  $x$  and  $y$  displacements,  $\delta_x$  and  $\delta_y$ , and the phase difference, for each filter  $G_j$ , is given by

$$\begin{aligned} \Delta \psi^j(x, y) &= \psi_2^j(x, y) - \psi_1^j(x, y) \\ &= \psi^j \left( x + \frac{\delta_x}{2}, y + \frac{\delta_y}{2} \right) - \psi^j \left( x - \frac{\delta_x}{2}, y - \frac{\delta_y}{2} \right) \\ &\approx \psi_x^j(x, y) \delta_x + \psi_y^j(x, y) \delta_y \quad j = 1, \dots, N_F. \end{aligned} \quad (13)$$

The above set of equations can be solved at point  $(x, y)$  if the response of more than one filter is not singular. In general, if the results of several filters are available, the solution is chosen that minimizes the square error,

$$\epsilon^2(\mathbf{x}) = \sum_{j=1}^{N_F} w^j(\mathbf{x}) \left[ \Delta \psi^j(\mathbf{x}) - \nabla \psi^j(\mathbf{x}) \cdot \boldsymbol{\delta}(\mathbf{x}) \right]^2. \quad (14)$$

The two unknowns  $\delta_x$  and  $\delta_y$  are the solution of the linear system

$$\frac{\partial \epsilon^2(\mathbf{x})}{\partial \delta_x(\mathbf{x})} = 0, \quad \frac{\partial \epsilon^2(\mathbf{x})}{\partial \delta_y(\mathbf{x})} = 0. \quad (15)$$

The weight factors  $w^j(\mathbf{x})$  measure the confidence in the response of the filter. A weight is set to zero if the corresponding filter has singular response. In the non-singular regions, we choose the weights to be proportional to the product of the left and right singularity detectors,

$$w^j(\mathbf{x}) = \begin{cases} (T_S - S_1^j(\mathbf{x})) (T_S - S_2^j(\mathbf{x})) & \text{if } S_1^j(\mathbf{x}) < T_S \\ & \text{and } S_2^j(\mathbf{x}) < T_S \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

To be non singular, a filter response  $\nabla \psi^j(\mathbf{x})$  has to be close to the tuning frequency  $\mathbf{k}_0^j$ . Therefore, to minimize the extension of singular areas, the filters should be chosen to cover as well as possible all directions in the range  $[0, \pi]$ .

#### 4 Implementation on a parallel DSP system

The aim of this work is the realization of a vision processing system for robot navigation, capable of producing dense disparity maps and flow fields in a time between 100 and 200 ms. The requested processing speed limits the size of the images and therefore the accuracy of the estimated fields – the relation between image size and performance will be discussed in Sects. 5 and 6. The system has to be mounted on a mobile platform, therefore size and power consumption has to be limited.

Phase-based techniques are local and domain decomposition is the most direct way to parallelize calculations. From a theoretical point of view, SIMD machines would be especially suited for this kind of computation. However, these are still large, high-cost machines not fit for “embedded” applications. We have chosen to implement the vision system on a parallel DSP system because of its flexibility and the availability of standard hardware for image acquisition and standard software tools for program development.

Furthermore, we point out that the locality of phase-based calculations is determined by the “receptive field” of the Gabor filters. The size of the filter must be large enough to include disparities that can easily reach significant portions of the image. Therefore, for an effective parallelization, in which communications between processors are minimized, each node of the machine should work on a subimage which is at least as large as the size of the Gabor filter. This implies that granularity cannot be too fine for a given image size. A coarse-grained parallel machine in which nodes are DSPs well matches the computational characteristics of the algorithm, if each processor takes care of a “receptive field”.

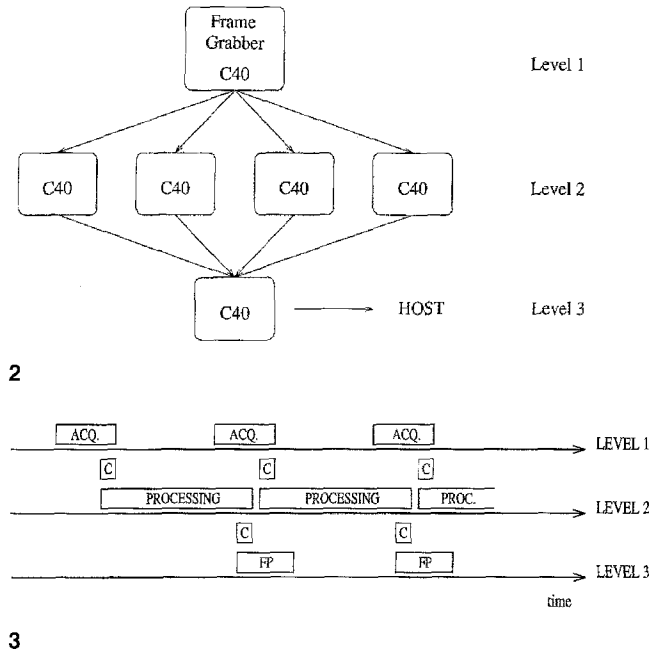


Fig. 2. The DSP system is a three-level pipeline: 1. image acquisition, 2. distributed processing, and 3. combination of partial results

Fig. 3. Schematization of the computational procedure. Processes are indicated as: *ACQ* = image acquisition, *PROCESSING* = image processing, *C* = communication, and *FP* = further processing

#### 4.1 The parallel DSP system

The vision processing system consists of five NEL TIM-40 modules and a NEL TIM-40 CFG frame grabber [12]. Each module has a TI-C40 DSP (50 MHz) and 2 or 5 MB of memory. The frame grabber is a true color grabber with a  $512 \times 512 \times 32$  dual-ported framestore with an onboard C40 (40 MHz). The TIMs are plugged in to two NEL PC-2M motherboards with PC bus.

The TI-C40 [16] is a DSP with Harvard architecture produced by Texas Instruments. Its peak performance is 340 MOPS and 60 MFLOPS at 50 MHz. Each module has six communication ports with independent DMA devices and maximum transfer rate of 20 MB/s. With these characteristics, the C40 can be utilized as the basic building block for a MIMD parallel machine with complex topology.

The configuration of the vision processing system, sketched in Fig. 2, is a three-stage pipeline: 1. image acquisition, 2. distributed elaboration, and 3. combination of partial results.

The visual signals from the two cameras are captured simultaneously by the frame grabber. Different resolutions (sampling frequencies) can be selected. The onboard processor converts the signal into two image matrices that are sent concurrently to the four processors of level 2. The computational procedure is subdivided among the four processors to exploit the algorithmic or the geometrical parallelism. These processors perform independently and are therefore not interconnected. Individual results are sent at approximately the same time to the processor of level 3, where they are recombined and made available for further processing or visualization (see Fig. 3).

Since each communication port has its own DMA processor, it is possible to overlap computation and communication. All processes are organized in threads and the acquisition, the communication and the computation tasks are assigned to different threads. To minimize the *reaction time* of the system, i.e. the time the system takes to respond to a stimulus, input images are not stored in multiple communication buffers. The acquisition of a new image is enabled only when the processing of the previous one is almost completed.

The programming language used for the algorithm implementation is an ANSI C with parallel libraries developed by 3L Ltd. [1] that allows simple management of parallelism. Most of the code was written using an optimized vectorial library developed by Sinectionalysis Inc. [15].

For the computation of the disparity or velocity field, the processor in the third level is only used to combine partial calculations and make the result available for operations that are based on global features of the fields. This processor also provides the necessary interface with other components of the robot.

## 5 Disparity estimation

The stereo configuration with parallel optical axis is considered<sup>1</sup>. Position disparity is a shift along the  $x$  direction of the image matrices,  $I_1(x, y)$  and  $I_2(x, y)$ . The signals from the two cameras are synchronized and images are captured simultaneously. To reduce acquisition time, only one field of each frame is used. The image pair, of size  $256 \times 256$ , is reduced to size  $128 \times 128$  by subsampling (by averaging) four pixels into one.

The simplest and most flexible way to parallelize the algorithm is domain decomposition. Each image is divided into four subimages  $32 \times 128$  that are sent to the four processors in level 2. Each processor receives two stripes, one from the left image, the other from the right image. The computation performed by each processor is schematized as follows.

1. Each processor performs a one-dimensional FFT of the rows of the corresponding subimages of size  $32 \times 128$ .
2. The Fourier transforms of the two subimages are multiplied by a selected Gabor filter and anti-transformed. The results are the complex functions  $C_k(x)$ ,  $k = 1, 2$ .
3. Phases and amplitudes are computed,

$$\psi_k(x) = \text{atan} \frac{\text{Im } C_k(x)}{\text{Re } C_k(x)},$$

$$\rho_k(x) = \sqrt{(\text{Im } C_k(x))^2 + (\text{Re } C_k(x))^2}, \quad (17)$$

4. The derivatives of phases and amplitudes along the  $x$  direction are computed using a two-point formula.
5. Neighborhoods of singular points are detected according to formula (8), with  $T_S = 1.2$ . A signal-to-noise constraint on the filter energy is added,  $\rho(x) > 0.05 E_M$ ,

<sup>1</sup> The two cameras are optically adjusted using maximum resolution,  $512 \times 512$ . Computation is sensitive to calibration errors, however, image pre-averaging (4 pixels  $\rightarrow$  1) reduces the sensitivity of the system to possible degradation of the calibration precision.

**Table 1.** Processing times for disparity estimation (ms)

Image size	Capture time $T_A$	Prep. time $T_P$	FFT time $T_T$	Cycle time $T_F$	Comm. time $T_C$	Total time for $N_F = 1$
$128 \times 128$						
Averaged	55	22	11	$N_F \times 60$	10	158
Subsampled	35	3	11	$N_F \times 60$	10	119

where  $E_M$  is the maximum of the filter's energy over the whole image.

6. Position disparity is computed using formula (7), with the averaged phase derivative,  $(\psi'_1(x) + \psi'_2(x))/2$ , at the denominator.
7. The partial results from the four processors are collected and patched together by the processor in level 3. At this point, the total disparity maps are available for further processing.

If several filters are used, points 2–5 have to be repeated for each filter. In the following, this cycle is referred to as the *processing cycle* for one filter.

### 5.1 Processing times

Processing times are displayed in Table 1. They are divided as follows:  $T_A$  is the acquisition time for a pair of stereo images,  $T_P$  is the time required to prepare the images for processing,  $T_T$  is the time to perform direct Fourier transform on the stereo pair,  $T_F$  is the processing cycle time for one filter, and  $T_C$  is the time delay due to communications between processors ( $T_C$  is usually larger than the physical communication time because of synchronization delays).

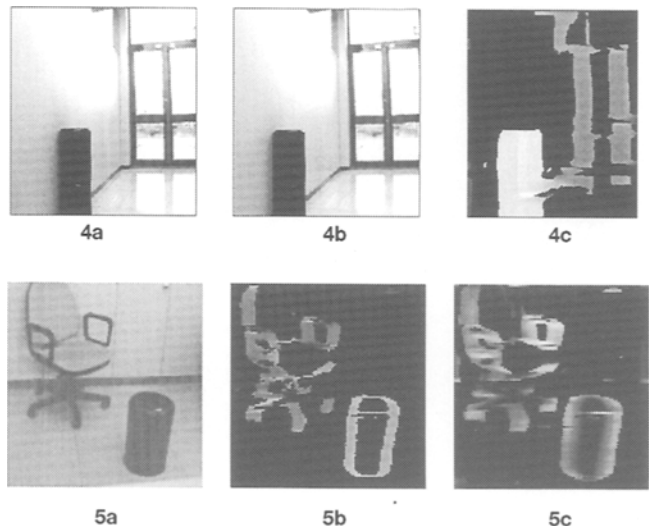
Acquisition, communications and computations are partially overlapping processes because they are performed in parallel. Acquisition time  $T_A$  is an average value because it depends on the synchronization between camera signals and processing cycle (since the camera synchronism is independent from the DSP synchronism, variations of  $T_A$  may be as large as 20ms). Processing time  $T_F$  may vary, usually not more than 5%, depending on the type of image and illumination. Variations of the other times are not more than 2%.

The total processing time (reaction time) for one image, i.e. the time required to produce the disparity map upon presentation of an image, is given by

$$T_{TOT} = T_A + T_P + T_T + N_F T_F + T_C.$$

The processing time  $T_T + T_F$  for one filter, for  $128 \times 128$  images, is 71 ms on our system, while on a SUN SPARC 20 (60 MHz) the same computation takes about 730 ms with a not fully optimized code. The time difference between two successive images is approximatively given by the processing time of level 2,  $\Delta t \approx (T_P + T_T + N_F T_F)$  (see Fig. 3). This means that images are processed at a rate of  $1/\Delta t$  images per second.

Figures 4 and 5 show some disparity maps obtained by applying one filter of wavelength  $\lambda = 128/10$  to images of size  $128 \times 128$ . Disparity maps are represented as gray level images; singular regions are black.



**Fig. 4.** **a** and **b** Stereo pair. **c** Disparity map, the filter's width,  $\sigma = \lambda/2.1$ , is 6 pixels

**Fig. 5.** **a** Image of a chair and a dustbin. **b** Disparity map,  $\sigma = 6$  pixels. **c** The filter's energy  $\rho$  in the non-singular regions that survive constraint (9)

### 5.2 Algorithm's performance

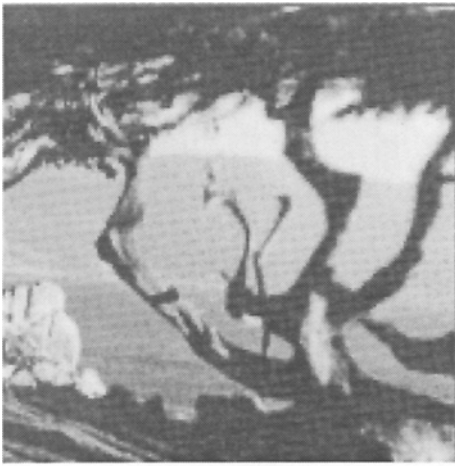
The algorithm's performance depends on the choice of filter width and on image quality. Referring to the single image, the filter width  $\sigma$  has to be chosen to better detect the image's contrast variations. In fact, for  $\beta \approx 1$ , the real and the imaginary parts of the Gabor filter act as edge detectors, for lines and steps, respectively.

If the width is chosen appropriately, a single filter produces a dense disparity map. Most of the singular regions are due to low signal-to-noise ratio, and are well detected by the constraint defined by Eq. (8), see Fig. 5c. These vast singular regions shrink or expand as the filter width  $\sigma$  is decreased or increased, respectively (see Fig. 9).

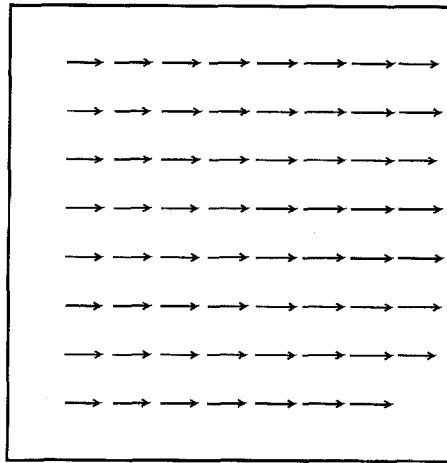
A filter cannot be expected to detect disparities larger than its width. If the disparity is too large, the visual features contained in the left region are different from those contained in the right region. Furthermore, since phase differences are defined modulo  $2\pi$ , a wraparound effect (aliasing) occurs when disparities are greater than half of the tuning wavelength,  $\lambda/2$ .

A quantitative analysis of the performance is obtained by applying the algorithm to standard images with known disparities. Here, we adopt the definition of error<sup>2</sup> and use test images provided by [3]. Two images of the translating tree sequence are used as test images – one of the images

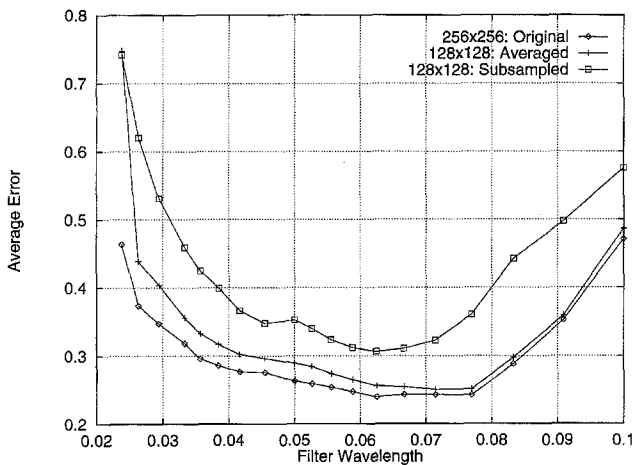
<sup>2</sup> In [3] velocity vectors  $\mathbf{v} = (u, v)$ , in pixel/frame units, are written as 3D vectors in space-time,  $\mathbf{v} = (u, v, 1)$ , in units of (pixel, pixel, frame). The error is defined as the angular deviation between the estimated ( $\mathbf{v}_e$ ) and the true ( $\mathbf{v}_t$ ) flow field,  $\psi_E = \arccos(\mathbf{v}_e) \cdot \mathbf{v}_t$  — in the formula vectors are normalized.



6



7



8

and the computed flow field is displayed in Fig. 6 and Fig. 7 – note the original  $150 \times 150$  images are zero-padded to size  $256 \times 256$  to handle FFT. The filter’s bandwidth is one octave,  $\beta \approx 1$ , and the singularity threshold is  $T_S = 1.2$ . Figure 8 displays the average error as a function of the filter width. On the x-axis we read the filter wavelength  $\lambda = 2.1 \sigma$  (in fractions of the image size), while the y-axis measures the average error. The tree curves represent, going downward, the result obtained using as input the original images of size  $256 \times 256$ , the same images reduced to size  $128 \times 128$  by averaging four pixels into one, and reduced to size  $128 \times 128$  by subsampling, respectively. The density, i.e. the ratio between the non-singular points and the total number of points is illustrated in Fig. 9. The figure shows that lowering resolution degrades algorithm performance. However, the effect can be reduced by lowering resolution by averaging. In our system, a size reduction of two in each dimension reduces processing time by one fourth: while computation on a  $256 \times 256$  image would take about 500 ms, the averaging process takes only 39 ms more than subsampling (see Table 1) and significantly improves system performance.

**Fig. 6.** One of the images of the Translating Tree sequence

**Fig. 7.** The disparity field of the Translating Tree computed for an input image of size  $128 \times 128$  (averaged)

**Fig. 8.** Behavior of the algorithm’s error as a function of the filter wavelength,  $\lambda = 2.1 \sigma$ . The curves correspond to different versions of the input image

For fixed resolution, the filter width is an important factor. The graphs in Fig. 8 provide evidence that there is an “optimal” choice that minimizes the error. The minimum is generated by two competing factors: large filters have a blurring effect and extend the disparity estimated to a large area around the target point, narrow filters are more accurate and can detect fine features, but if their width is too small the signal-to-noise ratio becomes too low for proper estimation of the disparity. Note that the error is computed only at non-singular points, it measures how precise the estimation of the disparity is at points where the algorithm result is accepted. As evidenced in Fig. 9, the number of points where the algorithm produces an estimate of the disparity increases with the filter width. In particular, it is important that the filters are large enough to detect the largest disparities in the image.

To improve performance, a set of filters, with different peak frequencies, may be applied to a single image and their results combined [10, 14]: the outputs of wide filters are used in regions where disparities are large, while the output of the narrow filters are used in regions of small disparities.

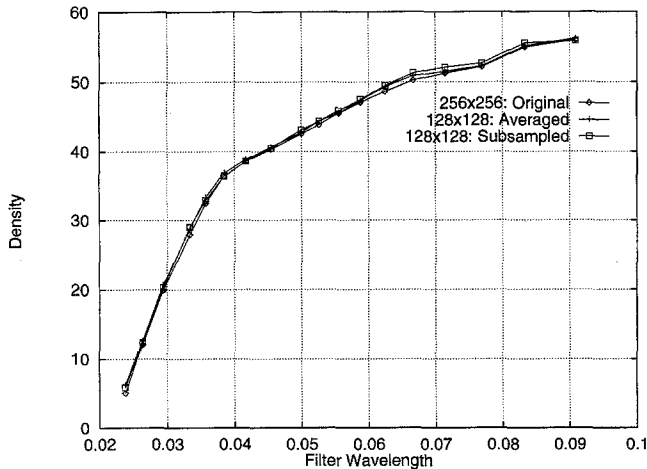


Fig. 9. Density of non-singular points as a function of filter wavelength,  $\lambda = 2.1 \sigma$ , measured in fraction of the image's linear size

Alternative techniques, in which images are locally shifted according to the disparity value computed can be devised [2, 11]. However, these methods require excessive processing times and are therefore not pursued.

For real-time applications, such as autonomous navigation, processing speed becomes very important and a compromise between speed and result accuracy has to be reached. For this reason, we use one filter, whose width is adjusted dynamically. The filter width is changed depending on the disparity values detected in the previous images. The width of the filter is increased when disparities close to the maximum are detected. Vice versa, if detected disparities are in the range of a narrower filter, the width is decreased. This mechanism ensures that the system responds appropriately if objects in the visual field are approached continuously. To avoid sudden appearances that go undetected, a mechanism that switches to the widest filter upon abrupt changes in the scene is added.

## 6 Optical flow estimation

The estimation of the optical flow involves the computation of local 2D shifts between two images,  $I_1 = I(t_1)$  and  $I_2 = I(t_2 = t_1 + \Delta t)$ , taken at successive times. The time difference,  $\Delta t$ , depends on the processing speed.

Since the problem is bidimensional, a set of directional filters must be used (see Fig. 1). With respect to the 1D algorithm, the processing time for a single filter is increased because 2D FFTs and 2D derivatives of amplitudes and phases need to be computed. Furthermore, for a fixed peak frequency  $k_0$ , a set of filters,  $N_F \geq 2$ , has to cover the direction spectrum. Finally, a linear system that integrates the responses of the individual filters has to be solved.

To achieve real-time processing, captured images, of size  $128 \times 128$ , are reduced to size  $64 \times 64$  by averaging four pixels into one. Each image is divided into four slightly overlapping patches, so that each processor of level 2 works on subimages of size  $32 \times 32$ .

Each processor in level 2 performs the entire computation cycle for  $N_F$  directional filters. For each direction,

Table 2. Processing times (ms) for estimation of the optical flow.  $T_A$  is an average value. Variations of  $T_F$  are not more than 5%, depending on the image. Variations of other times are not more than 2%

Acq. time $T_A$	Prep. time $T_P$	FFT time $T_T$	Cycle time $T_F$	Comm. time $T_C$	Image size
47	2	3	$N_F \times 21$	4	$64 \times 64$

Table 3. Average total processing times (ms)

Number of filters	$N_F = 2$	$N_F = 3$	$N_F = 4$	$N_F = 5$
Processing time	96	117	138	158

the processing results are combined with those obtained for the previous image and memorized for the successive step. The phase differences  $\Delta\psi^j = \psi_2^j - \psi_1^j$ ,  $j = 1, \dots, N_F$ , the phase gradients  $\nabla\psi_{av}^j = (\nabla\psi_1^j + \nabla\psi_2^j)/2$ , and the confidence vectors  $w^j$  are inserted in equation (14) to compute the displacements  $\delta_x$  and  $\delta_y$ . The confidence weights,  $w^j(\mathbf{x})$ ,  $j = 1, \dots, N_F$  are defined by Eq. (16). The linear system (15) can be solved if at least two confidence weights are different from zero.

The last processor patches together the subimages from the four processors into a  $64 \times 64$  velocity matrix, in which at each non-singular point  $(x, y)$  the values of the  $x$  and  $y$  displacements are given. The velocity at each point is then obtained dividing the displacement by the time interval  $\Delta t$ .

### 6.1 Processing times

Processing times are shown in Table 2. Time  $T_A$  is the acquisition time for one image of size  $128 \times 128$ . Time  $T_P$  is the time to reduce the image size to  $64 \times 64$ .  $T_T$  is the time to perform a  $32 \times 32$  FFT.  $T_F$  is the complete processing cycle time for each filter and  $T_C$  is the delay time due to communication among processors.

The total processing time (3) for one image, i.e. the time required to produce the velocity field upon the presentation of an image, is given by

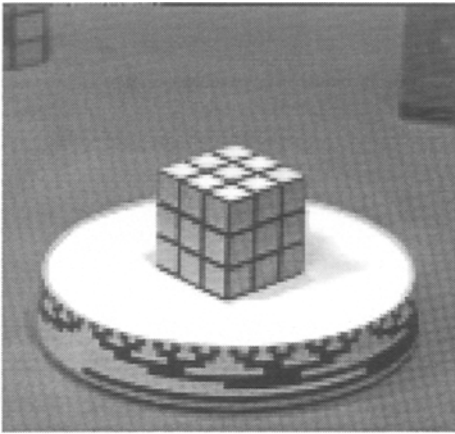
$$T_{TOT} = T_A + T_P + T_T + N_F T_F + T_C.$$

For example, the four-filter computation, without acquisition, takes 90 ms on our system, while on a SUN SPARC 20 (60 MHz) the same computation takes about 1200 ms with a not fully optimized code.

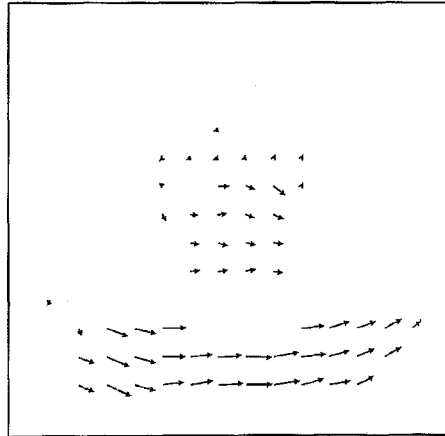
The time difference between two successive images is approximatively given by the cycle time of level 2,  $\Delta t \approx (T_P + T_T + N_F T_F)$  (see Fig. 3), so that  $1/\Delta t$  images per second are processed.

The angular width of a set of directional Gabor filters is chosen to cover as well as possible the orientation spectrum in the range  $[0, \pi]$ . For  $N_F$  filters, centered at  $0, \pi/N_F, 2\pi/N_F, \dots, (N_F - 1)\pi/N_F$ , the angular width of the Gaussian envelope is set to  $\tau_2 \approx \frac{\pi}{2N_F} k_0$ . This corresponds to a width of  $\tau_2 = 2.0 \nu_0$  for  $N_F=4$ . The radial bandwidth of the filters is one octave,  $\beta = 1$ ,  $\tau_1 = 2.1 \nu_0$ .

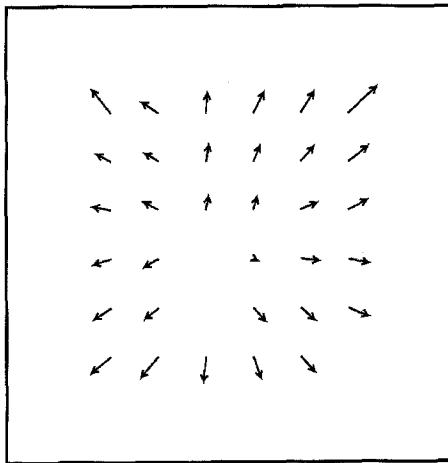
A test of optical flow estimation on the rotating Rubik cube (Fig. 10) is shown in Fig. 11. It displays a subsampled arrow representation of the velocity field for an input image of size  $60 \times 64$  obtained through the reduction, by averaging, of the original  $240 \times 256$  image.



10

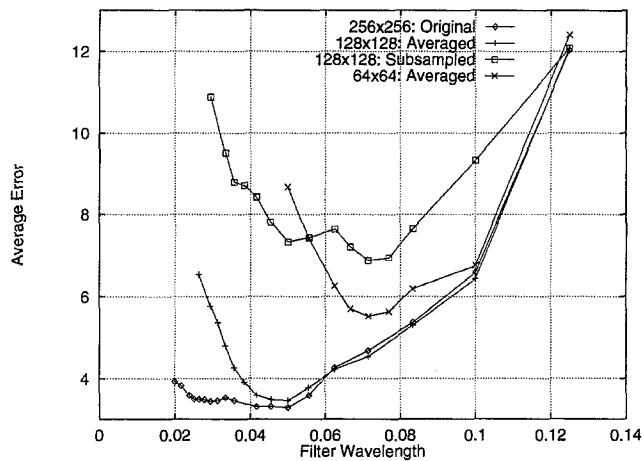


11



12

Fig. 10. The rotating Rubik cube

Fig. 12. Flow field for the Diverging Tree, computed using an input image of size  $64 \times 64$  (averaged)

13

Fig. 11. The velocity field of the Rubik cube, computed using an input image of size  $60 \times 64$  (averaged)Fig. 13. Behavior of the algorithm's error as a function of the filter wavelength,  $\lambda = 2.1 \sigma$ , measured in fraction of the image's linear size. The curves correspond to different versions of the input image

## 6.2 Algorithm's performance

Performance of the 2D algorithm is tested on the *Diverging Tree* sequence, Fig. 12 shows its flow field computed using a  $64 \times 64$  (averaged) image. Figure 13, obtained using four oriented filters, illustrates the performance as a function of the filter's width, for different versions of input images. Each line represents the average error as a function of the filter width.

First of all, we note that the error is about four times larger than the error reported in [3]. This is because in [3] Gabor filters extend to the temporal dimension and the algorithm works on a stream of images. However, the 3D technique requires such a prohibitive amount of processing time (about 4 min on a SUN SPARC 20, 60 MHz) that it is not applicable to real-time computation with standard hardware.

To achieve real time with the chosen processing system, images have to be reduced to size  $64 \times 64$ . This means that the resolution has to be lowered or the visual field has to be restricted. Size reduction, as illustrated by Fig. 13, degrades

**Table 4.** Algorithm's performance using different numbers of oriented filters,  $N_F$

$N_F$	2	3	4	5	8
Average error	3.3	2.9	3.4	3.4	3.2
Standard deviation	2.0	1.7	2.4	2.2	2.2
Density	15 %	25 %	33 %	39 %	47 %

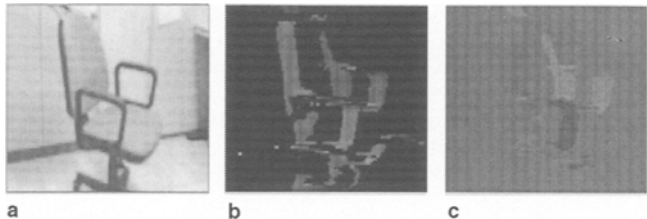
algorithm performance. The effect is reduced by lowering resolution by averaging.

Table 4 shows how the number of filters  $N_F$  influences system performance. The primary effect of increasing the number of filters is to increase the density of non-singular points. In fact, they cover more efficiently the frequency space (see Fig. 1).

Processing speed is a crucial factor for real-time applications and image size is reduced to gain speed. Regarding the effect of size reduction on the flow field estimation, we note that the error increase due to size reduction is compensated by a reduction of the time interval  $\Delta t$  between captured images. Table 5 (obtained using the Diverging Tree sequence)

**Table 5.** Comparison between the algorithm's performance using the average of the four pairs (18,19), (19,20), (20,21), and (21,22) of size  $128 \times 128$  and the pair of images (18,22) of size  $256 \times 256$ . The third row shows the results for the full-size image pair (19, 20)

Images	Average error	Standard deviation	Density
Average on 4 pairs	3.2	2.5	46 %
Pair (18,22)	4.1	2.1	25 %
Pair (19,20)	3.3	2.2	39 %



**Fig. 14.** **a** Image of a spinning chair ( $128 \times 128$ ). **b** Disparity map. **c** Gray-level image of  $v_x$ , the projection of the velocity vector on the  $x$ -axis; the medium gray tone (intensity=127) represents  $v_x = 0$ , whiter tones ( $127 < \text{intensity} < 255$ ) represent  $v_x > 0$ , and darker tones ( $0 < \text{intensity} < 127$ ) represent  $v_x < 0$

compares the error in the flow field for a pair of images  $I(t = 18)$  and  $I(t + 4 = 22)$ , and the averaged error for four pairs of images,  $I(18)$  and  $I(19)$ ,  $I(19)$  and  $I(20)$ ,  $I(20)$  and  $I(21)$ , and  $I(21)$  and  $I(22)$ , reduced to half the original size. The algorithm performance is better in the second case for two reasons: 1. the flow vectors are smaller when  $\Delta t = 1$ , and 2. the error is an average over four cases. The third row shows the error for the full-size image pair (19, 20),  $\Delta t = 1$ , and in this case performance is comparable.

From the qualitative point of view, the flow fields obtained from the reduced images reproduce the characteristics of the true flow fields well, if the features contained in the input images are large enough as in the examples shown in Figs. 11 and 12.

### 6.3 Simultaneous estimation of depth and motion

As was pointed out in the introduction, the computations of disparity map and optical flow are based on the same procedure. In the first case,  $\Delta(\psi)$  is the phase difference between the left and the right images,  $I_1(t)$  and  $I_2(t)$ , in the other it is the phase difference between successive images, for example  $I_1(t+1)$  and  $I_1(t)$ . Therefore, partial calculations obtained for disparity computation at time  $t$  can be used for the computation of the optical flow at time  $t + 1$ , if the same filters are used. As a by-product an edge representation of the image is also obtained.

A simplified module for estimation of disparity and 1D velocity maps was implemented. The same 1D filter is used in both computations. The results are the disparity map and the projection of the optical flow on the  $x$  axis (see Fig. 14). Referring to the definitions used for Table 1, the processing time  $T_F$  for images of size  $80 \times 128$  is of  $55 \pm 5$  ms. The time difference between images is  $\Delta T = 60 \pm 5$  ms. The reaction time of the system is  $T_{TOT} = 90 \pm 10$  ms.

## 7 Discussion

The choice of phase-based techniques for image matching is dictated by several occurrences. The phase-based differential technique is more robust than amplitude-based techniques for small deformations and contrast changes between pairs of images [5]. For the computation of the optical flow, this technique was shown to be more accurate than other methods [3]. The existence of an intrinsic confidence measure that estimates the reliability of computed results is an important property of the method. In a real-world environment the capability to know that the result computed is incorrect is of the utmost importance.

Another reason for the choice concerns the wide range of applicability and the flexibility of the phase-based technique. It can be applied for computation of disparity, optical flow, and image orientation. It does not rely on the choice of particular visual structure characteristics of a set of images. The method contains few parameters that can be easily adjusted to cope with changing inputs. Furthermore, there is physiological evidence that Gabor filters may model neural processing in the visual cortex [4].

One of the principal drawbacks of the phase-based technique is the high computational load. Each filter requires a complete independent computation cycle, therefore the processing time increases almost linearly with the number of filters and with the size of the image matrix. However, since computations are local, the algorithm can be parallelized efficiently.

Parallelization of computations was realized by domain decomposition. For the 1D algorithm this is the most straightforward and efficient way, since only one filter is used. Regarding the 2D case, two choices are possible: 1. to subdivide images in slightly overlapping patches and perform the complete computation on each patch, and 2. to assign to each of the processors a set of oriented filters and perform the corresponding computation on the total image. The first choice is penalized by border effects between patches, while the second requires the level 3 processor to perform the integration of the outputs of the different filters.

Since there are four processors in level 2, the second solution is efficient only for four oriented filters. The processing time and the quality of the results are practically the same for the two configurations when four oriented filters are used. We have described only the first way of parallelization, because in this configuration the choice of number of filters is more flexible.

In this work, the local nature of phase-based computations was exploited to realize a parallel vision processing system for depth and movement perception. Using a TI-C40-based DSP system with four computing processors, the system is capable of real-time computation of disparity maps for images of size  $128 \times 128$  and real-time computation of velocity field for images of size  $64 \times 64$ . Particular attention was given to the effect of image size reduction on flow fields estimation error.

The parallel system on which the algorithms were implemented, constitutes the minimal configuration that produces useful results in real time. This simplified vision system will be employed in autonomous navigation tasks. If more processors are available, it is reasonable to assume that initially

performance will scale almost linearly with the number of processors. However, due to the limited number of communication ports of each processor, the topology of the connections should be revised and an appropriate routing system should be included.

A larger number of processors may be employed to increase the image size and/or to increase the number of filters. Parallelization may exploit the independence of the processing of non-overlapping regions of the image and/or the independence of the computation cycles of different filters. On the basis of the results reported, the processing times necessary to compute disparity maps and optical flows in images of different size, using various numbers of filters, can be easily estimated.

*Acknowledgements.* This work was supported by the Japanese Ministry of International Trade and Industry as part of the Real World Computing Project [13].

## References

1. 3L Ltd. (1994) C40 Parallel CV1.1.1, March 1994
2. Anandan P (1989) A computational framework and an algorithm for the measurement of visual motion. *Int J Comput Vision* 2: 283–310
3. Barron JL, Fleet DJ, Beauchemin S (1994) Performance of optical flow techniques *Int J Comput Vision* 12(1): 43–77
4. Daugmann JG (1985) Representation issues and local filter models of two-dimensional spatial visual encoding. In: Rose D, Dobson VG (eds) *Model of the visual cortex*. Wiley, New York
5. Fleet DJ, Jepson AD (1993) Stability and phase information. *IEEE Trans PAMI* 15(12): 1253–1268
6. Fleet DJ, Jepson AD, Jenkin M (1991) Phase-based disparity measurement. *CVGIP: image understanding* 53(2): 198–210
7. Fleet DJ, Jepson AD (1990) Computation of component image velocity from local phase information. *Int J Comput Vision* 5(1): 77–104
8. Gabor D (1946) Theory of communication. *J Inst Electr Eng* 93 Part III: 429–457
9. Inoue H (1993) Vision-based robot behavior: testbed for real-world AI Research Proc IJCAI 1: 767–773
10. Jenkin M, Jepson AD (1988) The measurement of binocular disparity. *Computational processes in human vision*
11. Kopecz J, Von Seelen M, Bohrer S, Theimer W (1995) A neural architecture for visual information processing. *Int J Comput Vision* 16(3)
12. Nation Engineering Laboratory (1993) TIM-40CFG-RGB V2.0, User manual
13. Otsu N (1993) Toward flexible intelligence: MITI's new program of real-world computing. *Proc IJCAI* 1: 786–791
14. Sanger TD (1988) Stereo disparity computation using Gabor filters. *Biol Cybernetics* 1(59): 405–418
15. Sinectoanalysis Inc (1993) Optimized DSP vector library for TMS320C40. Reference manual
16. Texas Instruments Inc (1993) TMS320C4x, Users guide

**Bruno Crespi** graduated in physics at University of Trento, Italy. He received his Ph.D. in physics from the University of Illinois at Urbana-Champaign. Since 1990 he has been working at the Istituto per la Ricerca Scientifica et Tecnologica in Trento, Italy. His research interests include artificial vision, pattern recognition, associative memories, and neural networks.

**Gianni Di Caro** received his degree in physics from the University of Bologna, Italy. From 1992 to 1994 he worked at IRST (Istituto per la Ricerca Scientifica et Tecnologica), Italy. He is presently working at the Math. Dept. of the University of Trento. His research interests are parallel processing, robotics, optimization and neural networks.

**Franco Valentinotti** received his degree in physics from the University of Bologna, Italy. He performed the thesis work at IRST (Istituto per la Ricerca Scientifica e Tecnologica), Italy. He is working at ENEA (Ente Nazionale Energie Alternative) for the project HPCN (High-Performance Computing and Networking). His research interests are artificial vision, parallel processing, image processing, and cellular automata.