I. V. Toupitsyn
Siberian State Aerospace University named after academician M. F. Reshetnev,
Russia, Krasnoyarsk

THE FAST ALGORITHM OF INTERMEDIATE PERSPECTIVES RECONSTRUCTION ON A STEREOPAIR

Here we have considered the effective A. A. Lukianitsa algorithm of stereo-reconstruction. We have offered a modification of the particular algorithm, which allows increase in the speed of the algorithm functioning. We have also presented the results of the conducted experiments.

Keywords: stereo image, epipolar geometry, corresponding points, disparity.

One of the up-to-date trends of video information ideas is supposed to be the task of presenting dimensional data. Due to this, devices which are able to produce volumetric images (multiperspective monitors, projection systems) are becoming more and more widespread. For the proper functioning of these devices it is necessary to obtain a certain number of perspectives of the reproduced scene; the number of which is defined by a certain task. For instance, for watching videos 4–6 perspectives are usually enough for comfort. For serious applications such as 3D-tomography and radiography, graphical work stations CAD/CAM, the representation of presented settings (control tower, search, and rescue services), etc, one could require from 40 to 150 perspectives. There are some special devices providing from 2 (stereo-video cameras) to 8 perspective settings. 8 perspectives video cameras were used in a prototype multiperspective *NIKFI* television system, and it is quite difficult to imagine a video camera with a large number of perspectives.

Equally difficult is considered the recording and transmitting of such a signal through connecting channels. Consequently, there is a need to reconstruct a required number of perspectives – i.e. images which have been obtained with the help of a similar camera, if it had an intermediate position between cameras used in stereo filming. The majority of algorithms solve this problem on two levels: searching for the corresponding points, i.e. points at the images fitting to the same spot of a three dimensional space, and the approximation of intermediate perspectives on the basis of the reveled appropriate points. Usually, if all the corresponding points, found at the approximation of the intermediate perspectives cause no difficulties the primarily object of the intermediate perspective reconstruction is the search for corresponding points. As an example let us examine the functioning of a simple stereovision algorithm. It consists out of several steps:

1. We have selected the first (upper) epipolar line as a current line.
2. For the current epipolar line we have selected the pixel most left of the current pixel.
3. We have matched the current pixel of the left image to every pixel of the right image.
4. We have selected the most similar pixel from the right image and wrote it in the array of the corresponding pixels. The array of corresponding pixels represents a 2D integer-valued array; the index elements of which are the positions of the pixels on the left image and the meanings of the elements – are the positions of the corresponding pixels of the right image.
5. If the current pixel is not the last in the active row then the following pixel is getting the current pixel and the algorithm goes to item 3. Otherwise, to item 6.
6. If the active epipolar line is not the last line, then the following epipolar line becomes active and goes to item 2. Otherwise the algorithm stops functioning.

The functioning of a simple stereovision algorithm is presented as a logical diagram (as shown in fig. 1). The result of the algorithm functioning is an array of corresponding pixels. Then, using this array one can build a depth card or reconstruct the intermediate perspectives, i. e. images which could be obtained with a similar camera in case it took intermediate positions between the cameras, used in stereo filming.

The reconstruction of intermediate views according to the A. A. Lukianitsa method. Setting the task. Suppose we have a stereo pair – i.e. two images of the same setting fixed done with a stereo camera. Let's mark the image taken by the left camera, with letter L, and the image of the right camera – with letter R. Let's introduce the orthogonal coordinate system coordinated with the cameras: we direct axis X to the right, through the image centers, axis Y – downwards along the vertical image axis and axis Z adds them to the right three. The starting point of the coordinate shall be chosen in such a way that the images are in the Oxy plane, and image centers are symmetrical according to plane Oyz (fig. 2). This setting is registered by two cameras: L and R. We need to build an intermediate perspective M.

Say the center of the left image in the chosen coordinate system has the coordinates (x_L ,0 ,0), and the center of the right is – (x_R ,0 ,0). The task is that an image should be built in the plane Oxy with its center in a point (x_M, 0, 0), closely situated to the image which was received by the camera the optical axis of which would pass through point (x_M, 0, 0). The suggested method consists of two levels:

– the location of the corresponding points on the left and the right images;

– the approximation of the intermediate image on the base of conjugated points and the filling of the fields for which the correspondence is not single-digit [1; 2].

The search algorithm for corresponding points. Say every image contained $K \times N$ pixels on the horizontal and

vertical lines correspondingly. The task of finding the conjugated points is the following: for every pixel of the left image L_{ki}, $k = 1, \ldots, K$, $i = 1, \ldots, N$. We need to find pixel R_{kj}, $j = 1, \ldots, N$ at the right image corresponding with the same point of the registered settings.

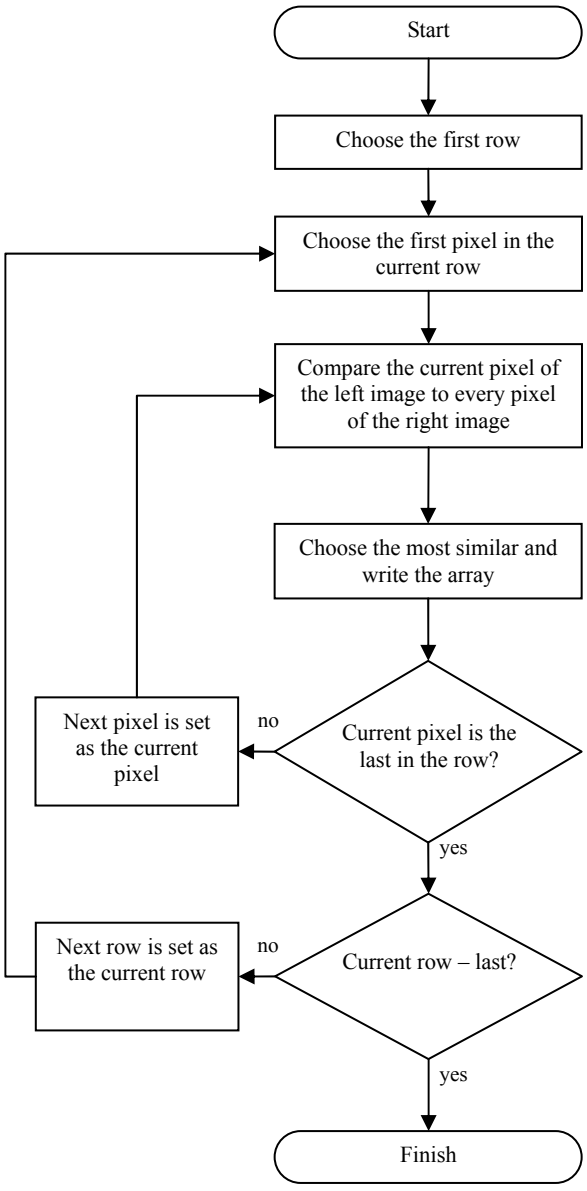

Fig. 1. The logical diagram of the stereovision functioning for a simple algorithm

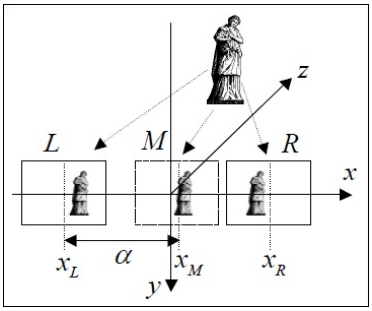

Fig. 2. Geometry of stereo filming

Suppose the cameras were fixed at the same horizontal line and have the same gain constant. Then the horizontal lines of the photo detectors coincide with the epipolar lines; that's why the conjugated point should be situated on the lines with the same number. This allows us to write down the following expression for calculation the epipolar lines:

$$L(k,i) \approx R(k, j + \Delta_j),$$

where Δ_j – is a shift, depending on the pixel number.

The finding of the corresponding points consists in the calculation of a displacement for all pixels. Further, for simplicity we shall omit the first index, because it has the same value for the compared images, and state that the process described below is executed for each pair of corresponding lines in the left and right images. To find expedient candidates for corresponding points we should compare some areas Ω in the environment of these pixels. It essentially increases the stability of the algorithm. Indeed, there may be differences in the images with a similar section of the scene under different angles, because of the mutual position of light sources and scenic elements. Besides, some areas can be visible for one camera and are invisible for another.

Let the compared area Ω have the following sizes $k \times n$, then in the environment of the j-th pixel on the right image we should scan a bigger sized area: $m \times l$, $m \geq k$, $l > n$. The distance function $d(i, j)$ is calculated during the scanning process. This function characterizes the proximity of images in area Ω_i of pixel $L(i)$ and in area Ω_j of pixel $R(j)$. The distance function is usually taken as a sum of differences for corresponding pixels intensities from Ω_i and Ω_j:

$$d(i, j) = \sum_{m=-k/n}^{k/2} \sum_{l=-n/2}^{n/2} \left| L(k+m, i+l) - R(k+m, j+l) \right|,$$

or as the comparing of the images from Ω_i and Ω_j gradients.

Practically, it is better to use a combination of these methods, because they have various efficiency for different types of images. Often, various researchers use a method of intensity comparison; however; when image processing with homogeneous vast objects, for example, a house wall, this method leads to the appearance of a plateau for function $d(i, j)$ that complicates a selection of corresponding points. In this case the gradient difference allows the considering of "thinner" image details.

To select the corresponding points from all possible pretenders we should meet two conditions. First of all, it's obvious that the index numbers of the corresponding points should minimize the total difference of the compared areas on the left and right images. Secondly, the offset value cannot decrease in the process of index j increase, i. e. $\Delta_j \geq \Delta_{j-1}$. This condition follows from the geometric properties of the stereo pair. Th next algorithm based on the dynamic programming method is offered to satisfy these conditions.

Let's introduce two accessory functions: real ψ and integer P. The first of these is necessary for the accumulation of total differences for compared areas. The second is reserved for the storage of appropriate indexes.

The dynamic programming algorithm consists of three steps:

a) Initialization:

$$\psi(i,1) = d(i,1),\ \ i = 1, ..., N;$$

$$P(i,1) = i, \qquad i = 1, ..., N.$$

b) Inductive passage:

$$\psi(i,j) = \min_{1 \le k \le i} \psi(k, j-1) + d(i,j),\ \ i = 2, ..., N,\ j = 1, ..., N,$$

$$P(i,j) = \arg\min_{1 \le k \le i} \psi(k, j-1), \quad i = 2, ..., N,\ \ j = 1, ..., N.$$

c) Reverse move:

$$J(N) = \arg\min_{1 \le i \le N} \psi(i, N),$$

$$J(j) = P(J(j+1), j+1),\ \ j = N-1, ..., 1.$$

The array of indexes for corresponding points J at which for any pixel of the left image $L(i)$ the corresponding pixel of the right image is described as $R(J(i))$ will be constructed in result. The dynamic programming algorithm is presented in fig. 3.

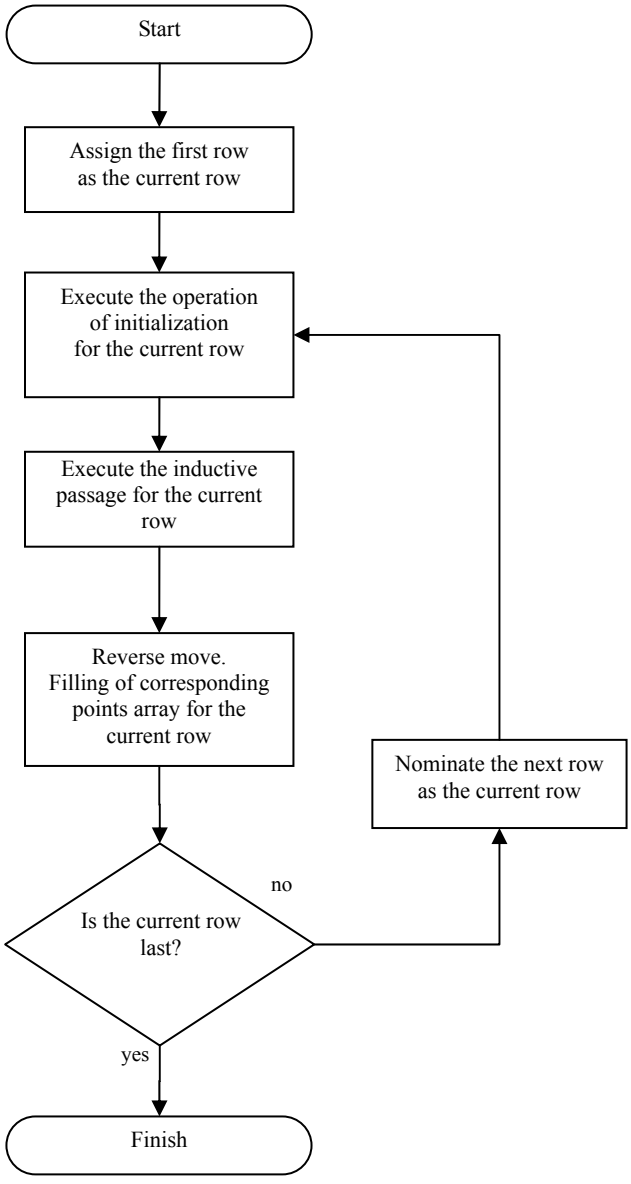

Fig. 3. The dynamic programming algorithm

This algorithm executes image processing line-by-line; this allows parallelizing calculations. It considerably speeds up stereo reconstruction procedures, but a consequence of progressive processing may be the effect of "combing". Unfortunately this algorithm does not offer any methods of reducing these effects.

Let's consider each step of the presented algorithm in detail. The initialization procedure serves for the definition of initial values of functions ψ and P. Function ψ is calculated as d function between the first pixel of the left image in the current line and each pixel of the right image in the current line. P function is calculated as pixel indexes of the right image.

The next step of the dynamic programming algorithm is the inductive passage. Here calculated are the subsequent values of functions ψ and P. Function ψ accumulates the total values of the compared areas. The total difference accumulation of compared areas increases the accuracy of finding corresponding points.

The last step of the dynamic programming algorithm is a reverse move. At this step the corresponding array point J is filled. The pixels of the right image, which have a minimal difference between each other, are selected as corresponding points for the pixels on the left image.

Intermediate view approximation

Let's introduce parameter $\alpha = \dfrac{x_M - x_L}{x_M - x_R}$, which characterizes the relative degree of proximity for the reestablished view on the left image. Further for all values of $i = 1,\ldots,N, j = 1,\ldots,N$ we will calculate:

$$k = [\alpha j + (1 - \alpha)J(j)],$$
$$M(i,k) = \alpha L(i,j) + (1 - \alpha)R(i, J(j)).$$

As we can see from these correlations some points of the intermediate view can be unfilled. This happens because the corresponding fragments of the three-dimensional object are visible for one camera, but invisible for the other. Let it be that in the reconstructed image in any i-th row pixels $j_0, \ldots, j_0 + m$ are unfilled. There are two cases:

– if $J(j_0 - 1) = J(j_0 + m + 1)$ then these pixels are visible only for the left camera and they are filled by proper values from the left image:

$$M(i,j) = L(i,j),$$
$$j = j_0, \ldots, j_0 + m;$$

– if $J(j_0 - 1) < J(j_0 + m + 1)$ then these pixels are visible only for the right camera and they are filled by respective values from the right image:

$$M(i,j) = L(i, J(j)),$$
$$j = j_0, \ldots, j_0 + m.$$

Increasing the speed of the algorithm. Though the speed of this algorithm is high enough, it can be insufficient for the solution of some tasks. Consequently, it's necessary to find a way of increasing the presented algorithm speed.

As a rule, in all stereo algorithms the CPU time for most of the part is occupied by the calculation of the distance function for the row pixels. To minimize the time spent on the distance function calculation it is necessary to simplify the function or decrease the quantity of pixels

for which it is calculated. One of the possible variants consists in decreasing the size of the frame around the selected pixels but this does not always give good results, and is not applicable for all images.

Apparently, from the aforesaid, the simplification of the distance function is not possible; therefore, it is necessary to reduce the number of computations for this function. The improvement of the discussed algorithm is linked with the introduction of an additional variable: maximal disparity $D_{\max}$. Maximal disparity is the maximal difference between the offset of a point in three-dimensional space on the left and right images. This variable is set by the user for each stereo pair. The value of this variable can be calculated if the location of the cameras and the distance from the stereo photography plane to the maximal distant object of the scene is known. Let's rewrite the expressions used on steps of the initialization, the inductive passage, and the reverse move; given the setting of maximum disparity in the following form:

1. Initialization:

$$\psi(i,1) = d(i,1), \quad i = 1,...,N;$$

$$P(i,1) = i, \qquad i = 1,...,N.$$

2. Inductive passage:

$$\psi(i,j) = \min_{D_{\max} \le k \le i} \psi(k, j-1) + d(i,j), \; i = 2,...,N, \; j = 1,...,N,$$

$$P(i,j) = \arg\min_{D_{\max} \le k \le i} \psi(k, j-1), \quad i = 2,...,N, \quad j = 1,...,N.$$

3. Reverse move:

$$J(N) = \arg\min_{D_{\max} \le i \le N} \psi(i, N),$$

$$J(j) = P(J(j+1), j+1), \quad j = N-1, ...,1.$$

In result of the entered changes, the speed of the algorithm considerably increases because now there is no necessity to compute the value of the distance function for each pair of pixels in a row. The presented algorithms were realized in *C*++ language. The realized experimental algorithms have been carried out (sce table). The time for locating corresponding points of one line has been measured. The distinctions in the results of algorithm operating have been evaluated. Comparison has been carried out on a test stereo pair with a 10 pixel width (fig. 4).

From table it is apparent that the modified algorithm has a higher speed. However, the quantity metric of truly defined corresponding pixels worsens by 2–6 % in comparison to A. A. Lukjanitsy's base algorithm [2].

In this article the stereo reconstruction algorithm of Lukianitsa A. A. had been overlooked. The speed of this algorithm was also estimated. Modifications, which increase the speed of this algorithm, were offered. The modified algorithm was realized in high-level programming language (*C*++). In result of the conducted experiments, the operation speed of the modified algorithm at insignificant deterioration of stereo reconstruction was registered.

References

1. Lukianitsa A. A. Effective Algorithm for Reconstruction of Intermediate Views from Stereo pair // Graphicon 2006 : Materials of Intern. conf. Novosibirsk, 2006. P. 15–18.

2. Bobick A. F., Intille S. S. Large occlusion stereo // Int. Journal of Computer Vision. 33(3). 1999. P. 181–200.

Experiment results

Stereo pair images width	Frame of compared areas size	Maximal disparity	Selected algorithm	Time of algorithm
472	5x5	–	Lukianitsa	0.343 s
		10	Modified	0.047 s
		20		0.047 s
716		–	Lukianitsa	1.311 s
		20	Modified	0.156 s
		40		0.265 s
3295		–	Lukianitsa	93.102 s
		40	Modified	5.304 s
		100		11.949 s

a *b*

Fig. 4. Test stereo pair :
a – left image; *b* – right image