Journal of ELECTRICAL ENGINEERING, VOL. 61, NO. 4, 2010, 247–251

REAL TIME MOTION DATA PREPROCESSING

Peter Benický — Ladislav Jurišica *

There is a lot of redundant data for image processing in an image, in motion picture as well. The more data for image processing we have, the more time is needed for preprocessing it. That is why we need to work with important data only. In order to identify or classify motion, data processing in real time is needed.

K e y w o r d s: motion, real time, identification, classification

1 INTRODUCTION

If a mathematical-physical model of the system cannot be created, abstract methods have to be used in order to solve the problem successfully. Image processing is such a problem. Recently, image processing is solved by artificial intelligence implemented by many kinds of neural networks that were inspired by human neural networks. However, computer computation of image data by any kind of neural networks costs much time and this approach cannot be used for real time application without optimization.

We should again return back to nature and try to see deeper how the human visual system works in real time. Let us have an example of an unknown photo album and try to analyze it picture by picture with 1 Hz frequency. We will not by able to process this information due to not having enough time for image analysis. However, if we see a movie in the cinema, where the frequency of image changing is much higher, real time processing of these data is possible. This is because after input data are being processed, image analysis and motion analysis work together as one system in the human visual system (Fig. 1). This is why, to be able to have motion or object recognition in real time based on data from a camera system, the recognition system should be more complex and should consist of the same subsystems as the human visual system, meaning image analysis and motion analysis. This paper considers fast motion data preprocessing for further motion analysis.

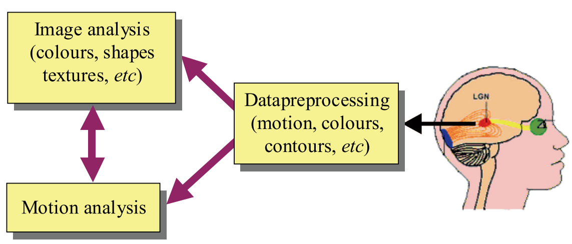

Fig. 1. Human visual system

Fig. 2. Motion detection using the low threshold (left), motion detection using a slow algorithm (right)

2 MOTION DETECTOR

The simplest motion detection is possible by two-frame comparison. If the frame difference in a specific pixel is higher than a defined threshold, then the specific pixel shows motion.

$$\Delta(x,y) = \left| I_{f^1}(x,y) - I_{f^2}(x,y) \right| . \tag{1}$$

However, to get motion detection with higher quality, the motion detector should be considered together with the quality of detector implementation. Please note that in case the motion detection algorithm is slow, the final result of motion detection will be with low quality even if the method of motion detection itself is good enough.

The presented motion detector is using fuzzy logic with empirical designed fuzzy rules. The detector is robust and the implementation of the algorithm for this detector can be optimized as will be described later in this section. Firstly, let us have two fuzzy sets named as "*big*" brightness change of the specific picture pixel and "*small*" brightness change. The fuzzy relevance functions for these fuzzy sets are defined by expressions (2) and (3), where a, b are experimentally defined parameters and $\Delta(x,y)$ is the brightness change of the specific picture pixel.

$$big\big(\Delta(x,y)\big) = \begin{cases} 0 & \text{for } \Delta(x,y) < a\,, \\ -\frac{\Delta(x,y)-a}{b-a} & \text{for } a \le \Delta(x,y) \le b\,, \\ 1 & \text{for } \Delta(x,y) > b\,, \end{cases} \tag{2}$$

* Slovak University of Technology in Bratislava, Faculty of Electrical Engineering and Information Technology, Institute of Control and Industrial Informatics, Ilkovičova 3, 812 19 Bratislava; peter@benicky.info, ladislav.jurisica@stuba.sk

ISSN 1335-3632 © 2010 FEI STU

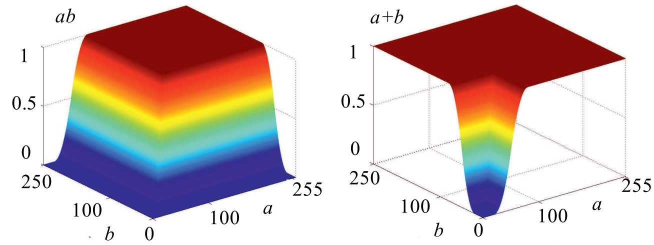

Fig. 3. Graph of fuzzy operations AND (left) and OR (right)

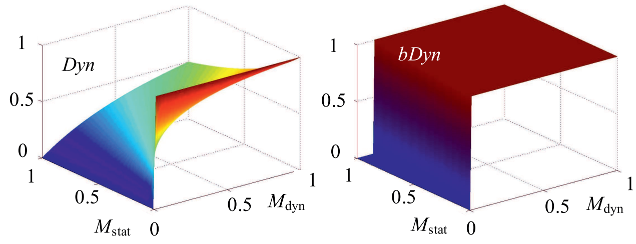

Fig. 4. Motion graph of function *Dyn* (*Mdyn*, *Mstat*) (left) and *bDyn* (*Mdyn*, *Mstat*) (right)

Fig. 5. Motion input in the row, the result of motion detection at the bottom. Slow motion in the left column, faster motion in the middle column and motion under direct strong light in the right column

$$small\big(\Delta(x,y)\big)=\begin{cases}1 & \text{for } \Delta(x,y)<a\,,\\ -\frac{1-\left(\Delta(x,y)-a\right)}{b-a} & \text{for } a\le\Delta(x,y)\le b\,,\\ 0 & \text{for } \Delta(x,y)>b\,.\end{cases} \tag{3}$$

For motion detection in a specific pixel we will consider, in addition to the brightness change of the specific pixel, also the brightness change of the pixels around. Thus we get a set of 9 brightness changes, let us label the set as *SP*.

$$SP=\{sp_1,sp_2,sp_3,sp_4,sp_5,sp_6,sp_7,sp_8\}\,. \tag{4}$$

The fuzzy rules were experimentally defined as below:

- If $\Delta(x,y)$ is *big* AND the third biggest element from set *SP* is *big*, then (x,y) is a dynamic pixel and we label it as $M_{\rm dyn}(x,y)$.
- If $\Delta(x,y)$ is *small* OR ($\Delta(x,y)$ is *big* AND the sixth smallest element from set *SP* is *small*), then (x,y) is a static pixel and it will be libelled as $M_{stat}(x,y)$.

Fuzzy operators AND and OR are defined as described in (5) and (6).

$$a \text{ AND } b=\min\{a,b\}\,, \tag{5}$$

$$a \text{ OR } b=\max\{a,b\}\,. \tag{6}$$

The surety of motion $Dyn(x,y)$ can be calculated by expression (7).

$$Dyn(x,y)=\frac{M_{Dyn}(x,y)}{M_{Dyn}(x,y)+M_{stat}\,(x,y)} \tag{7}$$

Finally, the pixel will be classified as motion if the surety of motion is higher than an experimentally defined threshold Γ. Let us have a binary value *bDyn* that indicates motion in pixel if the value is "1" as described in expression. As a result, black pixels will be in the picture with detected motion where *bDyn* for the pixel is "1".

$$bDyn=\begin{cases}0 & \text{for } Dyn(x,y)<\Gamma\,,\\ 1 & \text{for } Dyn(x,y)>\Gamma\,.\end{cases} \tag{8}$$

The fuzzy motion detection above was implemented in C++ language, where the algorithm was optimized by pre-calculated values stored in a one-dimensional array "look up" table. Please note that can be within the range of integers between 0 and 255 and there is no need to have the result of fuzzy operation AND, OR and values of $M_{dyn}(x,y)$, $Dyn(x,y)$ with higher precision than 1 decimal place. To provide higher performance, it is recommended to use the one-dimensional "look up" table since access into the one- dimensional array is faster than into

Fig. 6. Detected motion as the input data for edge detection (left), the result of the detected edges by a standard Laplace edge detector (right)

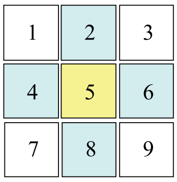

Fig. 7. Matrix of size 3×3 as reference template for method by masks

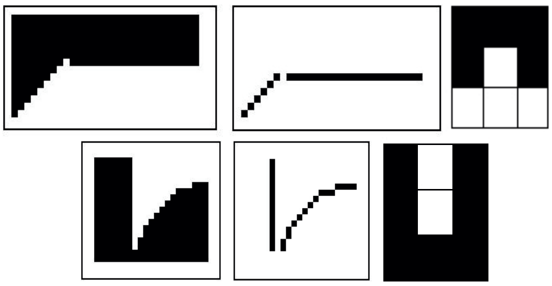

Fig. 8. Input image (left), detected edge (middle), appropriate patterns for fixing the detected edge (right)

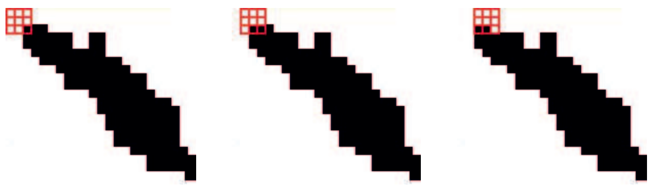

Fig. 9. The Process of searching for pattern matching in order to detect corrected edge

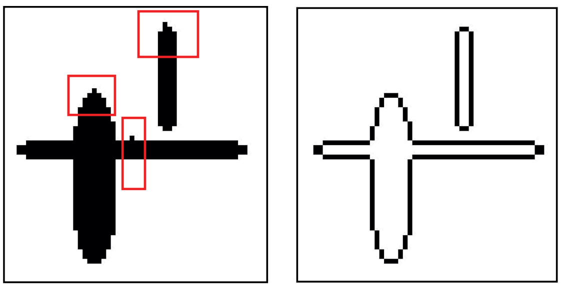

Fig. 10. Input image (left), detected edge with correction by pattern fixing images (right)

a multidimensional array. The results of the mentioned fuzzy motion detection with optimized algorithm are in Fig. 6.

We can see in Fig. 5 that the mentioned method is sensitive to a variety of object speeds and light intensity even when using a web camera with standard quality. An important parameter of this motion detector is also the computational time, where the average spent time is approximately 12 milliseconds on image of size of 100×100 pixels. However, the main goal of this approach is to have characteristic features that better describe the input data. The characteristics features could be for example the curvature of the detected motion area, or the curves themselves of the contours of the detected area (curves on the contours of detected motion area). For curves detection meaning line curvature that is higher than a specific threshold we need to have contours of the detected areas.

3 EDGE DETECTOR

Standard methods such as the Laplace edge detector (Fig. 6) do not provide contours that could be applicable for further curves detector methods. For this purpose, we need to have contours with a width of one pixel and without unreasonable line discontinuity. Therefore the solution must be based on real time processing, and the computer time processing needs to be considered as well.

Since the input data for the curves detector are binary (binary picture), the method based on masks was considered because this method is approximately 300 times faster than any standard method for edge detection [9]. Edge detection on a binary picture by the mask method is defined as follows: pixel "5" (Fig. 7) in the middle of matrix of size 3×3 is part of the edge (line) marked as black point if the pixel has at least one white pixel (white pixel is not part of the detected edge) among the neighbour pixels around ("2", "4", "6", "8") or there is at most one black pixel in all eight pixels around ("1", "2", "3", "4", "6", "7", "8", "9").

However, the mentioned rules do not provide contours with sufficient quality for further corner detection. To ensure a higher contours quality, we need to enlarge the group of rules. New rules were found empirically using a set of input binary images.

As an example, some new rules are shown in Fig. 8, where we can see that using a new fixing rule (using an appropriate pattern) the line is corrected and the line continuity is provided with a line of one pixel in width. By this approach, 89 fixing patterns were found. The algorithm of edge detection using pattern analysis is simple as described in Fig. 9, the matrix 3×3 is moving pixel by pixel. The matrix is then analyzed by pattern matching. If an appropriate pattern is found, then pixel "5" will be marked as the edge.

The result of using all found patterns is a more robust method for edge detection with a higher quality of detected edges of one pixel in width (Fig. 5).

Another advantage of this method, besides the quality of detected edge, is optimization during implementation

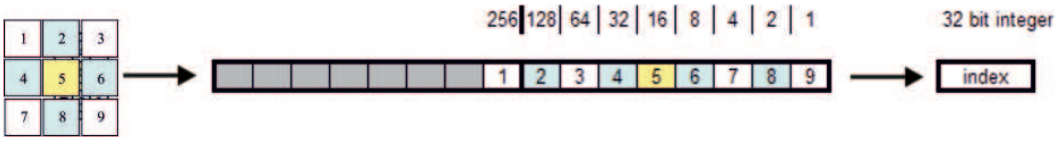

Fig. 11. Binary pattern image can be represented by one 32bit integer, where 9 bits are used

1	2	3	4	5	6	7	8	9	index
1	1	1	1	1	1	0	0	0	504
0	0	0	1	1	1	1	1	1	63
0	1	1	0	1	1	0	1	1	219
1	1	0	1	1	0	1	1	0	438
0	0	1	0	1	1	0	1	1	91
1	1	0	1	1	0	1	0	0	436
0	1	1	0	1	1	0	0	1	217
1	0	0	1	1	0	1	1	0	310
0	0	0	1	1	0	1	1	0	54
0	0	0	0	1	1	0	1	1	27
0	1	1	0	1	1	0	0	0	216
0	1	0	1	1	1	0	0	0	184
1	1	1	0	1	1	0	0	1	473
1	1	1	1	1	0	1	0	0	500
1	0	0	1	1	0	1	1	1	311
0	0	1	0	1	1	1	1	1	95
1	1	1	1	1	1	1	0	0	508

1	2	3	4	5	6	7	8	9	index
1	0	0	1	1	1	1	1	1	319
0	0	1	1	1	1	1	1	1	127
1	1	1	1	1	1	0	0	1	505
1	1	1	1	1	0	1	1	0	502
1	1	0	1	1	0	1	1	1	439
0	1	1	0	1	1	1	1	1	223
1	1	1	0	1	1	0	1	1	475
0	0	0	1	1	0	1	1	1	55
1	1	0	1	1	0	1	0	0	436
1	1	1	0	1	1	0	0	0	472
0	0	1	0	1	1	0	1	1	91
0	0	0	1	1	0	1	1	1	55
1	1	1	1	1	0	0	0	0	496
0	1	1	0	1	1	0	0	1	217
0	0	0	0	1	1	1	1	1	31
1	0	0	1	1	0	1	1	0	310
0	0	0	0	1	0	1	1	1	23

1	2	3	4	5	6	7	8	9	index
1	0	0	1	1	0	1	0	0	308
1	1	1	0	1	0	0	0	0	464
0	0	1	0	1	1	0	0	1	89
0	1	0	1	1	1	1	1	1	191
1	1	0	1	1	1	1	1	0	446
1	1	1	1	1	1	0	1	0	506
0	1	1	1	1	1	0	1	1	251
0	1	0	0	1	1	1	1	1	159
1	0	0	1	1	1	1	1	0	318
1	1	1	1	1	0	0	1	0	498
0	1	1	1	1	1	0	0	1	249
0	1	1	0	1	1	1	0	1	221
1	0	0	1	1	1	0	1	1	315
1	0	1	1	1	0	1	1	0	374
1	1	1	1	1	0	0	0	1	497
1	1	0	1	1	0	1	0	1	437
1	1	1	0	1	1	1	0	0	476

1	2	3	4	5	6	7	8	9	index
1	0	1	0	1	1	0	1	1	347
0	0	1	1	1	0	1	1	1	119
0	0	1	1	1	1	0	1	1	123
0	1	0	1	1	0	1	1	1	183
1	1	0	1	1	1	1	0	0	444
1	1	1	0	1	1	0	1	0	474
1	1	1	1	0	1	0	0	0	488
1	1	0	1	0	0	1	1	0	422
0	0	0	1	0	1	1	1	1	47
0	1	1	0	0	1	0	1	1	203
1	0	1	1	0	1	1	1	1	367
1	1	1	0	0	1	1	1	1	463
1	1	1	1	0	1	1	0	1	493
1	1	1	1	0	0	1	1	1	487
1	1	1	0	0	1	0	1	1	459
1	1	1	1	0	1	1	0	0	492
1	1	0	1	0	0	1	1	1	423

1	2	3	4	5	6	7	8	9	index
0	0	1	1	0	1	1	1	1	111
0	1	1	0	0	1	1	1	1	207
1	0	0	1	0	1	1	1	1	303
1	1	1	1	0	0	1	1	0	486
1	1	1	1	0	1	0	0	1	489
1	1	1	1	0	0	0	1	1	483
0	0	1	1	0	1	1	1	1	111
1	1	0	1	0	0	1	1	1	423
1	1	1	1	0	1	1	0	0	492
0	1	1	1	0	1	1	1	1	239
1	1	0	1	0	1	1	1	1	431
1	1	1	1	0	1	1	1	0	494
1	1	1	1	0	1	0	1	1	491
1	0	1	1	0	1	1	1	1	367
1	1	1	1	0	0	1	1	1	487
1	1	1	1	0	1	1	0	1	493
1	1	1	0	0	1	1	1	1	463

1	2	3	4	5	6	7	8	9	index
1	1	1	0	1	1	1	0	1	477
1	0	1	0	1	1	1	1	1	351
1	0	1	1	1	0	1	1	1	375
1	1	1	1	1	0	1	0	1	501

Fig. 12. Fixing patterns, 9 bits information on account of the patterns and its index

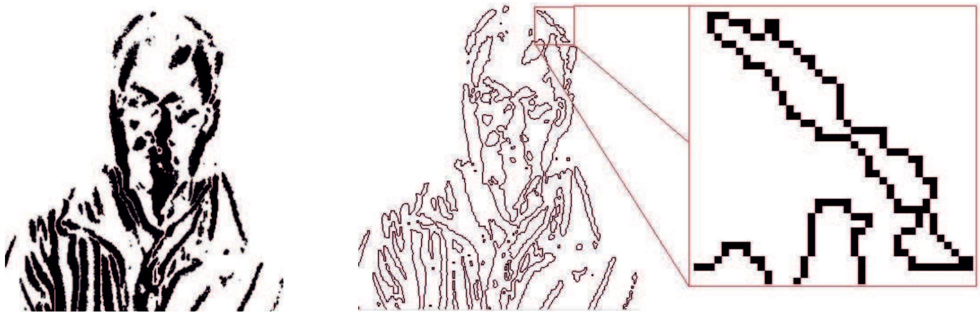

Fig. 13. Input image (left), the result of detected edge with correction by references images on real input data (right)

Fig. 14. The Picasso's hand sketches made by one single line

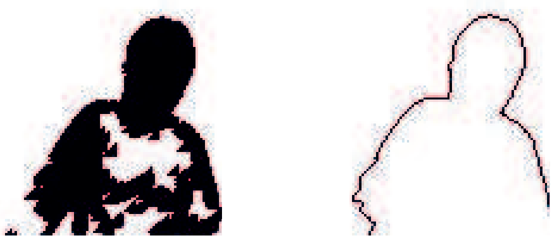

Fig. 15. Motion detection with multi-erosion (left), detected longest edge (right)

in order to provide a real time processing system. We can consider our 89 fixing patterns as 9 bit information per each pattern, meaning that this information can be covered by one 32 bit integer (in C++ language) for one fixing pattern.

Thus we have 89 integers that represent the true value meaning edge. To cover all combinations of 9 bit information we need to have an array of size $2^9 = 512$ (access to a one-dimensional array is faster than to a multidimensional array). Finally, when getting a matrix of size 3×3 from the input binary image, we simply convert the matrix into a 32 bit integer too and the information whether the pixel "5" is the edge or not will be provided by looking up into the array of integers.

This means, there are no mathematical-logical operations, since the look up tables of integers are pre-prepared before starting the entire algorithm. Finally, by the approach mentioned above we get the contours of movement with high quality (continual lines with one pixel of width) and fast since there are no mathematical-logical operations (memory processing only).

When seeing the edges in Fig. 13, it is obvious that data are not invariant on account of rotation. Such an approach can be useful for exact movement identification from a constant position of the observer. However, the minimum data needed for movement identification or

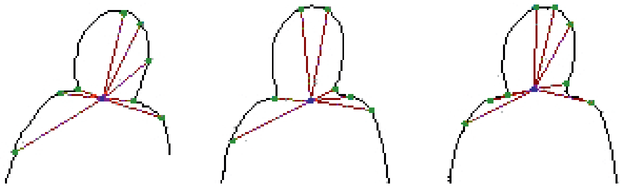

Fig. 16. Invariant motion descriptors defined as the distance between detected corner and motion object centroid

even object identification or classification is mentioned for example in Picasso's one line hand sketches (Fig. 14).

A similar approach can be calculated here as well. By multi-erosion of the input picture we can connect all moving parts together, parts that belong to the moving object. Having all the parts connected to one, then the longest edge describes the object in the same way as Picasso's hand sketches.

Having the longest edge of one pixel width that describes the moving object as one unit, rotation and zoom invariant characteristic features can be calculated. For curves detection the M2003 [11] algorithm was used. In order to make the invariant features, the object centroid is calculated so that the relative distances can be calculated by expression (9).

$$d_R[k] = \frac{d[k]}{\frac{1}{N}\sum_{k=1}^{N} d[k]} \tag{9}$$

where $d[k]$ is the corner, k is the absolute distance from the object centroid calculated by expression (10).

$$d[k] = \sqrt{\left[x[k] - x_c\right]^2 + \left[y[k] - x_y\right]^2},\ k = 0, 1, \dots, n-1\,. \tag{10}$$

Invariant motion descriptors that are created easily without heavy computational cost can be very useful for object identification or classification even when analyzing images (see Fig. 1), where the motion descriptors can help to faster identify objects during image analysis.

4 CONCLUSION

Rotation, zoom invariant motion descriptors based on curves of the longest edge of moving object calculated in real time are important data information for better object identification or classification. The approach presented in this paper is easy to implement and the computational cost is low due to working with the memory only for motion and edge detection.

References

[1] BATCHELOR, B. G.—HILL, D. A.—HODGSON, D. C. (Eds.) : Automated Visual Inspection, IFS Publications Ltd, UK, 1985.

[2] CHIN, R. T.—HARLOW, C. A. : Automated Visual Inspection: A Survey, IEEE Trans. Pattern Anal. Machine Intell. **PAMI-4** (Nov 1982), 557–573.

[3] MAITRE, G.—HUGLI, H.—TIECHE, F.—AMANN, J. P. : Range Image Segmentation Based on Function Approximation, published at ISPRS90, Zurich, Sep 1990.

[4] ELEWA, I. M.—SOLIMAN, H. H—ALSHENNAWY, A. A. : Computer Vision Methodology for Measurement and Inspection: Metrology in Production Area, Mansoura Eng. First conf. Faculty of Eng. Mansoura Univ., March 28-30, 1995, pp. 473–444.

[5] ALSHENNAWY, A. A. : Measurement and Inspection of Three Dimensional Objects Using Computer Vision System, Pd.D thesis, Mansoura University, Egypt, 2003.

[6] HOFMANN, H. D. : Application of Intelligent Measurements with Metrical Image Processing for Quality Control, presented at the 5th International Conference, PEDAC '92, Alexandria, EGYPT, Dec 1992.

[7] RUMMEL, P. : GSS - A Fast, Model-Based Gray-Scale Sensor System for Workpiece Recognition, Proceed. of 8th International Conf. on Pattern Recognition, Paris, France,, Oct 27-31, 1986, pp. 18–21.

[8] DARWISH, A. M.—JAIN, A. K. : A Rule Based Approach for Visual Pattern Inspection, IEEE Trans. on Pattern Analysis and Machine Intelligence **10** No. 1 (Jan 1988), 56–68.

[9] WAHDAM, A.-M. A. *et al* : Edge Detection if Binary Image using the Method of Masks, Sciencific bulletin **35** No. 3 (Sep 2000).

[10] internet http://www.cs.cf.ac.uk/Dave/C/node13.html .

[11] MARJI, M.—SIY, P. : A New Algorithm for Dominant Points Detection and Polygonization of Digital Curves, Pattern Recognition **36** No. 10 (2003), 2239–2251.

Received 16 December 2009

Peter Benický (Ing, PhD), received the Ing degree in 2005 and PhD degree in 2010, both in Electrical Engineering (Automation and Control) from the Slovak University of Technology, Faculty of Electrical Engineering in Bratislava. His main interests are the systems of enterprize resource planning and the video processing for robotic systems.

Ladislav Jurišica (Prof, Ing, PhD), received the Ing degree in 1964 and PhD degree in 1974, both in Electrical Engineering (Automation and Control) from the Slovak University of Technology, Faculty of Electrical Engineering in Bratislava. Since 1965 he was a research worker, then appointed Associate Professor in 1979. Since 1994 he is Full Professor at the Faculty of Electrical Engineering and Information Technology. His main interests are the systems of motion control, and the control of robotic systems. At present he is the leader of a research project in this field.