ISSN 1330-3651(Print), ISSN 1848-6339 (Online)
UDC/UDK 621.397.42:004.932.021

COMPRESSION METHODS FOR IMAGE PROCESSING IMPLEMENTATION INTO THE LOW CAPACITY DEVICES

Roman Slaby, Radim Hercik, Zdenek Machacek

Preliminary notes

This paper describes a development of systems with limited capacity and limited performance requires a specific approach in the implementation phase especially if the application has high demands which exceed the possibilities of the used hardware. The paper specifies the compression methods used for video signal processing from the camera, which is scanned and detected in the state consumption meter application. Used hardware Atmega1284p microprocessor is not suitable for video signal processing applications according to its parameters, but it can be realized using the proposed compression methods and specific implementation of algorithms. Design and verification of compression methods is important with respect to the best method choosing for detecting the video signal.

Keywords: *compression algorithm, compressed image, RLE, Atmega 1284p, flash memory, RAM, C#*

Metode kompresije za implementaciju obrade slike u uređaje malog kapaciteta

Prethodno priopćenje

Ovaj rad opisuje razvoj sustava s ograničenim kapacitetom zbog čega je potreban specifičan pristup u fazi primjene posebno ako je primjena zahtjevna i nadilazi mogućnosti upotrebljenog hardvera. U radu se specificiraju metode kompresije korištene za obradu video signala iz kamere, koji se skenira i prepoznaje primjenom mjerača stanja potrošnje (state consumption meter). Korišteni mikroprocesor hardvera Atmega1284p nije prikladan za obradu video signala prema njegovim parametrima, ali se to može učiniti primjenom predloženih metoda kompresije i specifične implementacije algoritama. Razvoj i verifikacija metoda kompresije je važno kad se radi o izboru najbolje metode za prepoznavanje video signala.

Ključne riječi: *algoritam kompresije, komprimirana slika, RLE, Atmega 1284p, flash memorija, RAM, C#*

1 Introduction

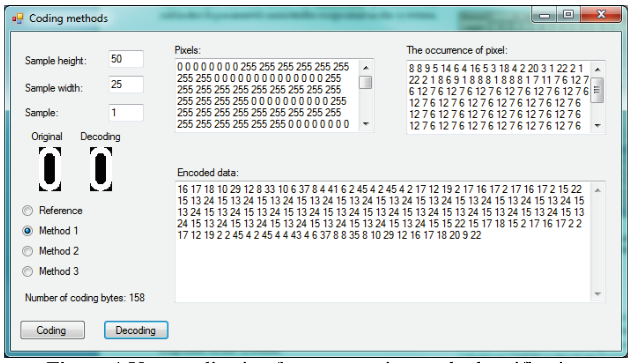
Compression algorithms are important in the case of limited resources and low capacity of data transmission. The task of compression is to reduce the bit rate or the bit size of the data matrix. Presented data compression methods are implemented for the application of image processing in microcontrollers, where the image patterns and a set of templates on the limited memory space must be stored. These stored data are usable for later image processing and correlation calculation. It is necessary to minimize the availability of memory space, because of the system dynamic range conservation, and because of the sufficient space for data structures processing.

There is a permanently removed part from the original information, thus lower image quality in the loss-making compression methods. The loss-less compression methods are less effective, but it is possible to make a complete reconstruction of the original information. Presented compression methods are implemented and tested in a microcontroller Atmega1284p for image patterns storing and image processing in consumption meter application. Evaluation of the compression methods was performed by statistical analyses.

2 Description of technical solution for implemented compression methods

Implementation of compression methods is realized with Atmega1284p microcontroller, where crystal frequency is 20 MHz. It contains internal 128 kB flash memory for program and 16 kB RAM memory. The external flash memory AT45DB with limited capacity 4 Mbit is connected via the SPI interface. The memory is organized into 8192 pages of 512 bytes with two input SRAM buffers, which allow simultaneous writing and reading from different parts of the FLASH memory. The Flash memory is operated in slave mode, and the microcontroller is the master device, which controls the communication.

Compression methods are implemented in a device, which is used to state recognition of the consumption meters. Each of consumption meters contains the digits 0 to 9. Because of different fonts of digits, the devices have to contain standard digit patterns for each of these fonts. Each image digit pattern is defined by the dimensions 25×50 pixels, because of the limited storage space and sufficient quality. The pattern is stored in binary format, which occupies 157 Bytes in uncompressed form. Compression methods implementation leads to significant savings in storage space with regard to the number of images and to the limited size of flash memory [1].



Sample height: 50
Sample width: 25
Sample: 1

Pixels:
```
0 0 0 0 0 0 0 0 255 255 255 255 255 255
255 255 0 0 0 0 0 0 0 0 0 0 0 0 0 0 255
255 255 255 255 255 255 255 255 255
255 255 255 255 0 0 0 0 0 0 0 0 0 0 255
255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 0 0 0 0 0 0 0 0
```

The occurrence of pixel:
```
8 8 9 5 14 6 4 16 5 3 18 4 2 20 3 1 22 2 1
22 2 1 8 6 9 1 8 8 8 1 8 8 8 1 7 11 7 6 12 7
6 12 7 6 12 7 6 12 7 6 12 7 6 12 7 6 12 7 6
12 7 6 12 7 6 12 7 6 12 7 6 12 7 6 12 7 6
12 7 6 12 7 6 12 7 6 12 7 6 12 7 6 12 7 6
12 7 6 12 7 6 12 7 6 12 7 6 12 7 6 12 7 6
```

Encoded data:
```
16 17 18 10 29 12 8 33 10 6 37 8 4 41 6 2 45 4 2 45 4 2 17 12 19 2 17 16 17 2 17 16 17 2 15 22
15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15
13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13
24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 13 24 15 15 22 15 17 18 15 2 17 16 17 2 2
17 12 19 2 2 45 4 2 45 4 4 43 4 6 37 8 8 35 8 10 29 12 16 17 18 20 9 22
```

( ) Reference
[✓ Method 1]
( ) Method 2
( ) Method 3

Number of coding bytes: 158

Figure 1 User application for compression method verification

Compressed data of these patterns are decoded and the original image is reconstructed during the recognition process. This image pattern is subsequently correlated with the captured image from the camera. Beside these patterns, the original color image from the camera is stored in memory AT45DB at a resolution of 640×480 pixels. It takes up to 921 600 bits, which is a quarter of the size of the whole memory. The memory also contains a process copy of the image and the history of recognized digits. Memory saving for the images patterns storage is more important than time saving in the process of decompression and reconstruction of the original image.

User application is developed in C# programming language and it is usable for testing the proposed algorithms and effectiveness comparison of the presented compression methods (Fig. 1) [2].

2.1 Design and implementation of method 1 based on RLE compression algorithm

Compression method 1 is based on the loss-less compression method RLE (Run Length Encoding). It is used to encoding only one Byte. The method is designed on the reduction of repeated sequences of digits. Each sequence of encoded data is formed by one Byte, where the first part (7 bits) indicates the number of repetitions and the second (1 bit) indicates current value represented by white or black color (Fig. 2).

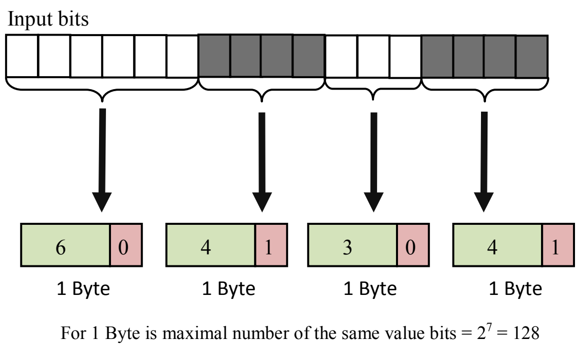

Input bits
6
0
4
1
3
0
4
1
1 Byte
1 Byte
1 Byte
1 Byte
For 1 Byte is maximal number of the same value bits = 2^7 = 128

Figure 2 Principles of method based on RLE algorithm

All of the identical pixel elements are calculated to one Byte in the process of pattern coding. In counting process of elements occurrence, there is computed only one row of the image pattern matrix, because of the easier reconstruction of the original pattern.

The compression process flowchart is shown in Fig. 3. The compression process consists of two cycles, where each row is gradually passed and the algorithm compares the value of the current element with the value of the previous element.

The variable *number* contains the elements number of the same value. There is stored value of the variable *number* and the value of the previous element to the compressed Byte, when the sequence of the identical elements is completed or the next row is started. Maximal sequence length of identical elements occurrence is defined by 7 bits, which corresponds to the maximal number $D_{\max}$ given by algorithm

$$D_{\max} = 2^7 = 128 . \quad (1)$$

There is a new Byte stored on the following position, when exceeds the maximum length sequence of identical elements occurrence. The compression method can be ineffective for inadequate data, because the size of the RLE coded data can also be double in comparison to the non-compressed data.

In the case of compression digits characters the modified RLE method is effective because the images with a small number of color changes have got the minimum number of Bytes. This encoding is suitable because of easy data reconstruction. In the method implementation each row is defined by a given number of Bytes, which represent elements values changes [3, 4].

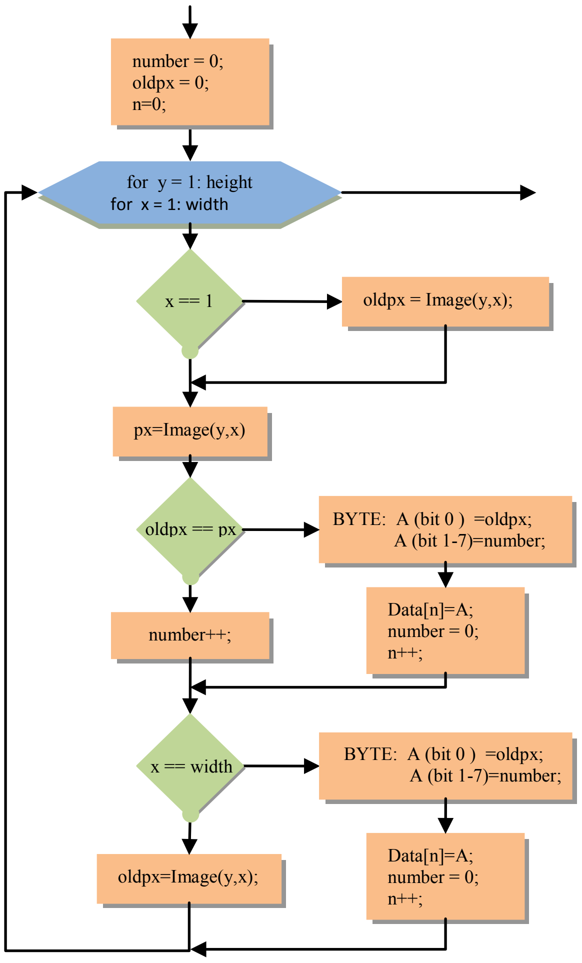


Figure 3 Flowchart of method based on RLE algorithm

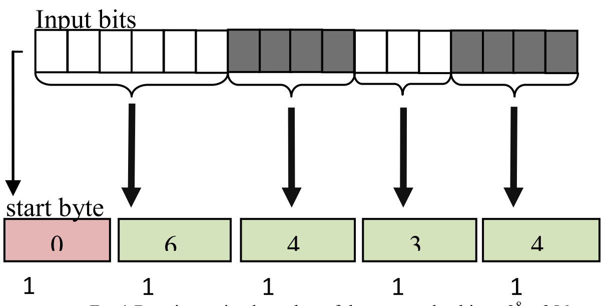

Input bits
start byte
0
6
4
3
4
1
1
1
1
1
For 1 Byte is maximal number of the same value bits = 2^8 = 256

Figure 4 Principles of method based on bit maps

2.2 Design and implementation of method 2 based on algorithm of bit maps

The compression method 2 is based on algorithm of bit maps and it is the loss-less compression method. The principle of method is based on the storing of the length of the identical elements occurrence sequence. The method includes a one start Byte, which contains information of the initial element value of the compressed image (Fig. 4).

The implementation of the compression method is shown in the flowchart (Fig. 5). The initialization of variables is the first step, where the data position is marked in the field, where the following compressed Byte will be stored. The variable contains the number of identical pixel elements sequence.

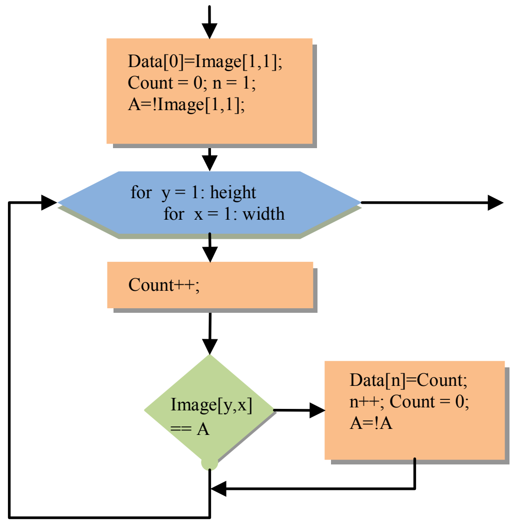


Figure 5 Flowchart of method based on algorithm of bit maps

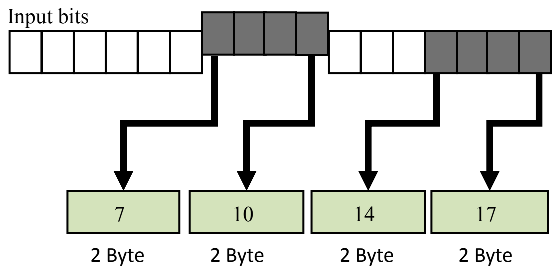
Maximal sequence length of identical elements occurrence is defined by 8 bits, which corresponds to the maximal number $D_{\max}$ given by algorithm

$$D_{\max} = 2^8 = 256. \qquad (2)$$

Each Byte contains the length of the identical elements sequence and the next Byte contains the length of the following inverse elements sequence. The compression method could be ineffective according to the previous compression method 1 [4].

2.3 Design and implementation of method 3 based on algorithm of position detection

Method 3 is based on loss-less compression using the coordinate's position system of the identical pixel values. The principle of the method is to store the coordinates of the starting position and the final position of the identical elements sequence equal to value (Fig. 6).



For 2 Byte is maximal number of the coordinates value bits = 2^{16} = 65535

Figure 6 Principles of method based on algorithm of position detection

The compression method implementation consists of processes of sequential searching the points of inverse value edge through all pixels in the image. The coordinates are stored to the compressed bytes in cases when value is changing to lg 1 or lg 0. In the same moment, the value of the indicator flag n is incremented (Fig. 7).

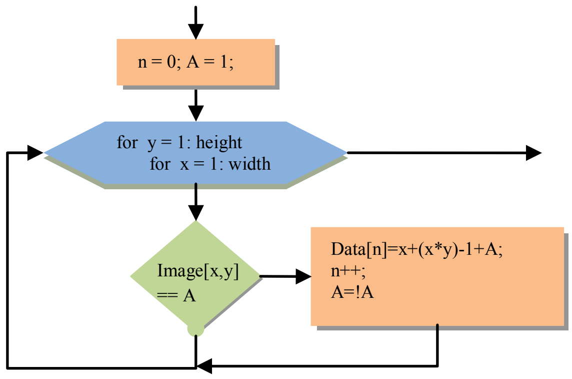


Figure 7 Flowchart of method based on algorithm of bit position

The method allows you to compress images with a maximal number of pixels $D_{\max}$ given by algorithm

$$D_{\max} = 2^{16} = 65536, \qquad (3)$$

here the Byte extension is very simple and universal for the maximal size changes of the encoded images.

3 Verification and testing of implemented methods for data compression in capacity limited applications

The presented compress methods were verified and the basic results are presented in the following Tab. 1. The verification was performed in the user application in the text above. There were applied three compress methods for defined digits series of 50×25 pixels size patterns. There were performed measurements and verification for each type of method. In the left column, there is given bytes B number N of the compressed image by the chosen method. On the top of the table, there is presented reference number N_{R} of bytes B of the non-compressed original image, which represents the chosen digits pattern [5]. The reference number N_{R} is the same value for all individual digits patterns and it is given by the presented algorithm

$$N_{\mathrm{R}} = \frac{H}{8}, \qquad (4)$$

here H is the number of all pixels of given image pattern, thus the calculation for this application is equal to

$$N_{\mathrm{R}} = \frac{50 \cdot 25}{8} = 156{,}25 = 157\ \mathrm{B}. \qquad (5)$$

In the right column, bytes reduction S is given which compares the compressed image to the non-compressed original image. The reduction S is given by

$$S = N - N_{\mathrm{R}}\,. \tag{6}$$

The reduction success P is the comparison of the compressed image bytes number N to the non-compressed image bytes number N_{R}.

$$P = \frac{N}{N_{\mathrm{R}}}\cdot 100\,. \tag{7}$$

Table 1 Example of results from described compression methods for image digits patterns storing

Non-compressed Original* Image N_R = *157 B				
Image	Used method	*N[B]*	*S[B]*	*P[%]*
0	Method 1	158	1	100,6
	Method 2	114	-43	62,3
	Method 3	224	67	129,9
1	Method 1	107	-50	53,3
	Method 2	98	-59	39,8
	Method 3	192	35	118,2
2	Method 1	133	-24	82,0
	Method 2	96	-61	36,5
	Method 3	188	31	116,5
3	Method 1	154	-3	98,1
	Method 2	122	-35	71,3
	Method 3	240	83	134,6
4	Method 1	145	-12	91,7
	Method 2	102	-55	46,1
	Method 3	200	43	121,5
5	Method 1	126	-31	75,4
	Method 2	112	-45	59,8
	Method 3	224	67	129,9
6	Method 1	157	0	100,0
	Method 2	138	-19	86,2
	Method 3	272	115	142,3
7	Method 1	132	-25	81,1
	Method 2	88	-69	21,6
	Method 3	176	19	110,8
8	Method 1	167	10	106,0
	Method 2	132	-25	81,1
	Method 3	260	103	139,6
9	Method 1	144	-13	91,0
	Method 2	112	-45	59,8
	Method 3	220	63	128,6

4 Conclusion

The presented article described the compression methods and implemented algorithms for image processing into the low capacity devices, especially the digit patterns recognition advanced system for consumption meters. The theoretical part of the compression methodology is complemented by practical examples and flow diagrams of conducted measurements of three chosen methods, which are designed for the digit recognition by image processing.

The application as the user interface for parameters setting and verification of compression methods was explained. The results in Table 1 show the rate of compression methods successful for each digit pattern type. The red marked digits in Table 1 represent non-successful compressed digits with respect to the compression method. There are presented additional or saved Bytes S and reduction success P from the number of Bytes evaluation for each compression method type.

From presented results, method 1 based on RLE compression algorithm is successful for all digits except digits with spherical character, method 2 based on algorithm of bit maps is the most successful method, method 3 based on algorithm of position detection is not applicable for digits. The comparison of compression methods and their design are very important for the effective system realization in developed low capacity devices.

Acknowledgments

The work was supported by project SP2012 /111 - Data collection and data processing of large distributed systems II. Grand – aided student R. Hercik, Municipality of Ostrava, Czech Republic.

5 References

[1] Gibson, J. D. Handbook of Image & Video Processing. – Academic Press London, 2000, p. 891.

[2] Machacek, Z.; Hercik, R.; Slaby, R. Smart user adaptive system for intelligent object recognizing. // Studies in Computational Intelligence, Berlin Heidelberg, 351(2011), pp. 197-206.

[3] Salomon, D. A Guide to Data Compression Methods, Springer, 2002, p. 304.

[4] Salomon, D. A Concise Introduction to Data Compression, Springer, 2008, p. 314.

[5] Holst, G.; Lomhein, T. CMOS/CCD Sensors and Camera Systems. // Society of Photo Optical, 2011, p. 392.

[6] Jan, J. Číslicová filtrace, analýza a restaurace signálů, nakladatelství VUTIUM, Brno, 2002, ISBN 80-214-1558-4 (2. uprav. vydání), pp. 287-308.

[7] Sovka, P.; Polák, P. Vybrané metody číslicového zpracování signálů, vydavatelství ČVUT, Praha, 2003, ISBN 80-01-02821-6, pp. 89-94.

[8] Chassaing, R.; Reay, D. Digital Signal Processing and Applications with the TMS320C6713 and TMS320C6416 DSK, John Wiley & Sons, Inc. New Jersey 2008, ISBN 978-0-470-13866-3, pp. 319-353.

Authors' addresses

Roman Slaby
Faculty of Electrical Engineering and Computer Science
VŠB - Technical University of Ostrava
17 Listopadu, 708 33 Ostrava-Poruba
Czech Republic
Tel.: +420 59 732 3497
E-mail: roman.slaby@vsb.cz

Radim Hercik
Faculty of Electrical Engineering and Computer Science
VŠB - Technical University of Ostrava
17 Listopadu, 708 33 Ostrava-Poruba
Czech Republic
Tel.: +420 59 732 3497
E-mail: radim.hercik@vsb.cz

Zdenek Machacek
Faculty of Electrical Engineering and Computer Science
VŠB - Technical University of Ostrava
17 Listopadu, 708 33 Ostrava-Poruba
Czech Republic
Tel.: +420 59 732 3497
E-mail: zdenek.machacek@vsb.cz